

**SISTEMA GRÁFICO INTERATIVO PARA  
ANÁLISE DE NUCLEAÇÃO E PROPAGAÇÃO  
DE TRINCAS**

**Leandro Lopes da Silva**

UNIVERSIDADE FEDERAL DE MINAS GERAIS  
ESCOLA DE ENGENHARIA  
CURSO DE PÓS-GRADUAÇÃO EM ENGENHARIA DE ESTRUTURAS

## **SISTEMA GRÁFICO INTERATIVO PARA ANÁLISE DE NUCLEAÇÃO E PROPAGAÇÃO DE TRINCAS**

Leandro Lopes da Silva

Dissertação apresentada ao Curso de Pós-Graduação em Engenharia de Estruturas da Escola de Engenharia da Universidade Federal de Minas Gerais, como parte dos requisitos necessários à obtenção do título de “Mestre em Engenharia de Estruturas”.

Comissão Examinadora:

---

Prof. Dr. Roque Luiz da Silva Pitangueira  
DEES - UFMG (Orientador)

---

Prof. Dr. Samuel Silva Penna  
DEES - UFMG (Coorientador)

---

Prof. Dr. Felício Bruzzi Barros  
DEES - UFMG

---

Prof. Dr. Evandro Parente Junior  
DEECC-UFC

Belo Horizonte, 29 de Fevereiro de 2016

S586s

Silva, Leandro Lopes da.

Sistema gráfico interativo para análise de nucleação e propagação de trincas [manuscrito] / Leandro Lopes da Silva. – 2016.

xxv, 218 f., enc.: il.

Orientador: Roque Luiz da Silva Pitangueira.

Coorientador: Samuel Silva Penna.

Dissertação (mestrado) Universidade Federal de Minas Gerais, Escola de Engenharia.

Apêndices: f. 154-210.

Bibliografia: f. 211-218.

1. Engenharia de estruturas - Teses. 2. Computação gráfica - Teses. 3. Método dos elementos finitos - Teses. I. Pitangueira, Roque Luiz da Silva. II. Penna, Samuel Silva. III. Universidade Federal de Minas Gerais. Escola de Engenharia. IV. Título.

CDU: 624(043)

*Quanto mais aumenta nosso conhecimento, mais evidente fica nossa ignorância.*

John F. Kennedy

*À minha namorada e à minha família.*

# Índice

Índice	iii
Lista de Tabelas	vi
Lista de Figuras	xi
Lista de Abreviaturas e Siglas	xii
Lista de Símbolos	xiii
Resumo	xxiii
Abstract	xxiv
Agradecimentos	xxv
<b>1 INTRODUÇÃO</b>	<b>1</b>
1.1 Objetivos	4
1.1.1 Objetivo Geral	4
1.1.2 Objetivos Específicos	4
1.2 Organização do Texto	5
<b>2 APLICAÇÕES GRÁFICAS INTERATIVAS DO INSANE</b>	<b>8</b>
2.1 Sistemas Orientados a Objetos	8
2.1.1 Classes e Objetos	9
2.1.2 Coleções de Objetos	9
2.1.3 Abstração	10
2.1.4 Encapsulamento	10
2.1.5 Modularidade	10
2.1.6 Herança	11
2.1.7 Polimorfismo	11
2.2 Padrões de Projeto de Software	12
2.2.1 Padrão <i>Model-View-Controller</i>	13
2.2.2 Padrão <i>Command</i>	14
2.2.3 Padrão <i>Observer</i>	16
2.3 Estrutura de Dados para Subdivisão Planar	17

2.3.1	Operadores de Euler . . . . .	21
2.4	Geração de Malhas . . . . .	22
2.4.1	Métodos Diretos ou Algébricos . . . . .	22
2.4.2	Métodos Indiretos ou de Equações Diferenciais . . . . .	26
<b>3</b>	<b>METODOLOGIA DE INTEGRAÇÃO DO SISTEMA GRÁFICO</b>	<b>27</b>
<b>4</b>	<b>EXEMPLOS DE MODELAGEM NO SISTEMA GRÁFICO</b>	
	<b>INTEGRADO</b>	<b>38</b>
4.1	Painel em “L” . . . . .	39
4.2	Chapa com Trincas em Modo I de Abertura . . . . .	44
4.3	Barragem Submetida à Carga Hidrostática . . . . .	50
4.4	Viga Pré-tensionada sob Flexão . . . . .	55
4.5	Viga com Pré-trinca sob Flexão . . . . .	60
4.6	Placa Anular com Carga na Borda Interna . . . . .	67
<b>5</b>	<b>MÉTODOS NUMÉRICOS PARA FISSURAÇÃO</b>	<b>71</b>
5.1	Métodos de Representação Geométrica . . . . .	72
5.2	Métodos de Representação Não Geométrica . . . . .	75
5.2.1	Métodos Constitutivos . . . . .	78
5.2.2	MEFG - Um Método Cinemático . . . . .	81
<b>6</b>	<b>NUCLEAÇÃO DE FISSURAS - TENSOR ACÚSTICO</b>	<b>85</b>
<b>7</b>	<b>MODELO CONSTITUTIVO DISCRETO</b>	<b>97</b>
<b>8</b>	<b>NUCLEAÇÃO E PROPAGAÇÃO DE TRINCAS E O NÚCLEO</b>	
	<b>NUMÉRICO DO INSANE</b>	<b>109</b>
<b>9</b>	<b>SIMULAÇÕES NUMÉRICAS</b>	<b>121</b>
9.1	Tração Direta . . . . .	122
9.2	Flexão em 3 Pontos . . . . .	126
9.3	Painel em “L” . . . . .	131
9.4	Compressão Diametral . . . . .	135
9.5	Viga em Balanço . . . . .	142
9.6	Cisalhamento em 4 Pontos . . . . .	146
<b>10</b>	<b>CONSIDERAÇÕES FINAIS</b>	<b>151</b>
10.1	Contribuições deste Trabalho . . . . .	151
10.2	Sugestões para Trabalhos Futuros . . . . .	153
<b>A</b>	<b>Processos Incrementais-Iterativos</b>	<b>154</b>
A.1	Controle de Carga . . . . .	157
A.2	Controle Direto de Deslocamento . . . . .	158
<b>B</b>	<b>Classe <i>Combinable</i></b>	<b>162</b>

<b>C Sistema Gráfico Integrado</b>	<b>174</b>
C.1 Módulo <i>Geometria</i> . . . . .	176
C.2 Módulo <i>Malha</i> . . . . .	177
C.3 Módulo <i>Atributos</i> . . . . .	186
C.4 Pós-Processador . . . . .	204
<b>D Classe <i>DiscontinuityByGFEM</i></b>	<b>206</b>
<b>Referências Bibliográficas</b>	<b>211</b>

# Lista de Tabelas

9.1 Resistência à compressão diametral . . . . .	141
--	-----



# Lista de Figuras

2.1	Arquitetura do INSANE (Penna, 2007). . . . .	13
2.2	Estrutura do padrão <i>Command</i> (Gamma et al., 1995). . . . .	16
2.3	Padrão <i>Observer</i> (Gonçalves, 2004). . . . .	16
2.4	Representação da aresta na estrutura “ <i>Half-Edge</i> ” (Mäntylä, 1987). . . . .	18
2.5	Estrutura “ <i>Half-Edge</i> ” (Adaptado de Mäntylä (1987)). . . . .	19
2.6	Estrutura “ <i>Half-Edge</i> ” modificada (Adaptado de Del Savio et al. (2004)). . . . .	20
2.7	Atuação dos operadores de Euler (Ferreira, 2008). . . . .	21
2.8	Malhas em Domínios Elementares (Gonçalves, 2004). . . . .	23
2.9	Malhas por Transformação de Coordenadas (Gonçalves, 2004). . . . .	23
2.10	Mapeamento Conforme (Gonçalves, 2004). . . . .	24
2.11	Mapeamento Isoparamétrico (Gonçalves, 2004). . . . .	24
2.12	Mapeamento transfinito (Gonçalves, 2004). . . . .	25
2.13	Decomposição de Domínio (Gonçalves, 2004). . . . .	25
2.14	Métodos Indiretos ou de Equações Diferenciais (Gonçalves, 2004). . . . .	26
3.1	<i>MVC-Observer</i> (Adaptado de Penna (2007)). . . . .	29
3.2	Estrutura hierárquica da classe <i>Combinable</i> . . . . .	30
3.3	Estrutura hierárquica das principais interfaces do sistema INSANE. . . . .	32
3.4	Estrutura gráfica do ambiente de análise bidimensional do INSANE. . . . .	33
3.5	Organização do núcleo numérico do INSANE. . . . .	36
4.1	Painel em “L”: Configuração (Penna, 2011). . . . .	39
4.2	Painel em “L”: Geometria. . . . .	41
4.3	Painel em “L”: Malha. . . . .	41
4.4	Painel em “L”: Condições de contorno. . . . .	42
4.5	Painel em “L”: Deformada. . . . .	43
4.6	Painel em “L”: Deformação $\varepsilon_1$ . . . . .	43

4.7	Chapa com trincas: Configuração. . . . .	44
4.8	Chapa com trincas: Geometria. . . . .	47
4.9	Chapa com trincas: Malha. . . . .	48
4.10	Chapa com trincas: Condições de contorno. . . . .	48
4.11	Chapa com trincas: Deformada. . . . .	49
4.12	Chapa com trincas: Tensão $\sigma_{yy}$ . . . . .	49
4.13	Barragem: Configuração (Penna, 2011). . . . .	50
4.14	Barragem: Geometria. . . . .	52
4.15	Barragem: Malha. . . . .	53
4.16	Barragem: Condições de contorno. . . . .	53
4.17	Barragem: Deformada no passo 500 ( $P = 0,33 MN$ ). . . . .	54
4.18	Barragem: Fator de carga $\times$ Deslocamento horizontal do nó sob $P$ . . . . .	54
4.19	Viga pré-tensionada: Configuração (Penna, 2011). . . . .	55
4.20	Viga pré-tensionada: Geometria. . . . .	57
4.21	Viga pré-tensionada: Malha. . . . .	58
4.22	Viga pré-tensionada: Condições de contorno. . . . .	58
4.23	Viga pré-tensionada: Dano na direção de $\varepsilon_1$ no passo 1500 ( $P_a = 1,81 kN$ ). . . . .	59
4.24	Viga pré-tensionada: Fator de carga $\times$ Deslocamento vertical do nó sob $P_a$ . . . . .	59
4.25	Viga com pré-trinca: Configuração (Penna, 2011). . . . .	60
4.26	Viga com pré-trinca: Geometria. . . . .	63
4.27	Viga com pré-trinca: Malha. . . . .	63
4.28	Viga com pré-trinca: Condições de contorno. . . . .	64
4.29	Viga com pré-trinca: Deformada no passo 200 ( $P = 7,4 kN$ ). . . . .	65
4.30	Viga com pré-trinca: Tensão $\sigma_{xx}$ no passo 200 ( $P = 7,4 kN$ ). . . . .	65
4.31	Viga com pré-trinca: Fator de carga $\times$ Deslocamento vertical sob $P$ . . . . .	66
4.32	Placa anular: Configuração (Adaptado de Penna (2011)). . . . .	67
4.33	Placa anular: Geometria. . . . .	68
4.34	Placa anular: Malha. . . . .	69
4.35	Placa anular: Condições de contorno. . . . .	69
4.36	Placa anular: Deformada. . . . .	70
4.37	Placa anular: Deslocamento $D_z$ . . . . .	70

5.1	Classificação dos métodos numéricos para fissuração que se baseiam no MEF (Adaptado de Ingraffea e Wawrzynek (2004)). . . . .	72
5.2	Descrição cinemática da zona de fissuração (Jirásek, 2001). . . . .	76
5.3	Estratégia de enriquecimento da nuvem $\varpi_j$ (Barros, 2002). . . . .	81
6.1	Corpo cortado por uma descontinuidade (Jirásek, 2007a). . . . .	86
6.2	Condições de continuidade de tensão e de deslocamento (Jirásek, 2007b). . . . .	87
6.3	Modos de falha no plano (a) I e (b) II (Adaptado de Jirásek (2007a)). . . . .	90
7.1	Corpo $\Omega$ cortado por uma descontinuidade $\Gamma_d$ (Wells e Sluys, 2001). . . . .	97
7.2	Sistema de coordenadas local da descontinuidade (Wolff, 2010). . . . .	104
8.1	Diagrama da classe <i>DiscontinuousEnrichment</i> . . . . .	110
8.2	Exemplo de investigação do valor do multiplicador de enriquecimento. . . . .	110
8.3	Intervenções na classe <i>GFemModel</i> . . . . .	111
8.4	Fluxograma do processo de nucleação e propagação de trincas. . . . .	112
8.5	Intervenções na classe <i>Element</i> . . . . .	114
8.6	Intervenções nas classes <i>AnalysisModel</i> e <i>Plane</i> . . . . .	114
8.7	Intervenções na classe <i>GFemElement</i> . . . . .	117
8.8	Intervenções nas classes <i>GFemParametric</i> e <i>GFEMPhysicallyNonLinear</i> . . . . .	120
9.1	Tração direta: Configuração (Wolff, 2010). . . . .	122
9.2	Tração direta: Deformada. . . . .	123
9.3	Tração direta: Tensão $\sigma_{yy}$ . . . . .	124
9.4	Tração direta: Trajetória de equilíbrio do deslocamento de controle $D_y$ . . . . .	125
9.5	Flexão em 3 pontos: Configuração (Penna, 2011). . . . .	126
9.6	Flexão em 3 pontos: Deformada. . . . .	127
9.7	Flexão em 3 pontos: Tensão $\sigma_{xx}$ . . . . .	128
9.8	Flexão em 3 pontos: Trajetória de equilíbrio do deslocamento do apoio direito. . . . .	129
9.9	Flexão em 3 pontos: Comparação dos resultados obtidos com 6, 12 e 18 elementos ao longo da altura. . . . .	130
9.10	Painel em “L”: Configuração (Winkler et al., 2004). . . . .	131
9.11	Painel em “L”: Deformada. . . . .	132
9.12	Painel em “L”: Dano na direção de $\varepsilon_1$ . . . . .	133

9.13	Painel em “L”: Trajetória de equilíbrio do deslocamento vertical máximo.	134
9.14	Padrão de dano observado por Winkler et al. (2004).	134
9.15	Compressão diametral: Configuração (Penna, 2011).	135
9.16	Compressão diametral: Deformada.	136
9.17	Compressão diametral: Tensão $\sigma_{xx}$ .	137
9.18	Compressão diametral: Deformada.	138
9.19	Compressão diametral: Tensão $\sigma_{xx}$ .	139
9.20	Compressão diametral: Trajetória de equilíbrio do deslocamento vertical do nó sob a carga.	140
9.21	Viga em balanço: Configuração.	142
9.22	Viga em balanço: Deformada.	143
9.23	Viga em balanço: Tensão $\sigma_{xx}$ .	144
9.24	Viga em balanço: Trajetória de equilíbrio do deslocamento de controle $D_y$ .	145
9.25	Cisalhamento em 4 pontos: Configuração (Penna, 2011).	146
9.26	Cisalhamento em 4 pontos: Deformada.	147
9.27	Cisalhamento em 4 pontos: Tensão $\tau_{xy}$ .	148
9.28	Cisalhamento em 4 pontos: Trajetória de equilíbrio do deslizamento tangencial relativo das extremidades da trinca (CMSD).	149
9.29	Cisalhamento em 4 pontos: Trajetória de equilíbrio do deslocamento vertical do nó sob a carga $P$ .	150
A.1	Algoritmo para métodos de controle (Fuina, 2009).	156
A.2	Processo incremental-iterativo com controle de carga (Fuina, 2009).	157
A.3	Processo incremental-iterativo com controle direto de deslocamento (Fuina, 2009).	159
A.4	Fatores de carga obtidos para os métodos de controle (Fuina, 2009).	160
C.1	Diálogo inicial do sistema INSANE.	175
C.2	Pré-processador gráfico interativo: Módulo <i>Geometria</i> .	176
C.3	Pré-processador gráfico interativo: Módulo <i>Malha</i> .	177
C.4	Mapeamento Transfinito: Divisão das arestas.	178
C.5	Mapeamento Transfinito: Discretização.	179
C.6	Geração Nodal Aleatória: Discretização.	180
C.7	Malha com base em Grade: Discretização.	180

C.8	Definição da dimensão média dos elementos no entorno do nó 1. . . . .	181
C.9	Discretização com concentração de malha. . . . .	182
C.10	Refinamento local da malha. . . . .	183
C.11	Diálogo de definição do <i>Combinable</i> e do modelo de análise. . . . .	184
C.12	Diálogo de definição dos elementos finitos. . . . .	185
C.13	Pré-processador gráfico interativo: Módulo <i>Atributos</i> . . . . .	186
C.14	Diálogo de definição do modelo constitutivo. . . . .	188
C.15	Diálogo de criação de materiais. . . . .	189
C.16	Material com uma lei de evolução de dano como um de seus parâmetros. . . . .	190
C.17	Diálogo de criação de degenerações. . . . .	191
C.18	Diálogo de criação de funções de enriquecimento polinomial. . . . .	192
C.19	Diálogo de criação de funções de enriquecimento de singularidade. . . . .	192
C.20	Diálogo de criação de descontinuidades. . . . .	193
C.21	Diálogo de criação de combinações de carregamentos. . . . .	194
C.22	Diálogo de criação de casos de carregamento. . . . .	195
C.23	Diálogo de aplicação de cargas nodais. . . . .	196
C.24	Diálogo de aplicação de cargas concentradas em arestas. . . . .	197
C.25	Diálogo de aplicação de cargas linearmente distribuídas em arestas. . . . .	198
C.26	Diálogo de aplicação de cargas de área. . . . .	199
C.27	Diálogo de atribuição de restrições aos deslocamentos nodais. . . . .	199
C.28	Diálogo de atribuição de rigidezes de mola aos deslocamentos nodais. . . . .	200
C.29	Diálogo de atribuição de pré-deslocamentos aos nós. . . . .	200
C.30	Diálogo de atribuição de funções de enriquecimento aos nós. . . . .	201
C.31	Diálogo de atribuição de degenerações aos elementos. . . . .	202
C.32	Diálogo de definição dos parâmetros da técnica de solução. . . . .	203
C.33	Pós-processador. . . . .	204
C.34	Diagrama de comunicação do sistema gráfico integrado. . . . .	205

# Lista de Abreviaturas e Siglas

MEF	Método dos Elementos Finitos
MEFG	Método dos Elementos Finitos Generalizados
INSANE	<i>INteractive Structural ANalysis Environment</i>
PU	Partição da Unidade
API	<i>Application Program Interface</i>
MVC	<i>Model-View-Controller</i>
GUI	<i>Graphical User Interface</i>
MEV	<i>Make Edge, Vertex</i>
XML	<i>eXtensible Markup Language</i>
UML	<i>Unified Modelling Language</i>
MEFX	Método dos Elementos Finitos Estendidos
CMSD	<i>Crack Mouth Sliding Displacement</i>

# Lista de Símbolos

## Capítulo 2

$V$	Números de vértices
$E$	Números de arestas
$F$	Números de faces
$S$	Números de sólidos (volumes independentes)
$H$	Números de furos passantes (gênero topológico)
$R$	Números de “ <i>loops</i> ” internos ( <i>ring</i> )

## Capítulo 4

$E$	Módulo de elasticidade
$\nu$	Coefficiente de Poisson
$\varepsilon_1$	Primeiro componente de deformação principal
$\lambda$	Parâmetro de entrada da função de enriquecimento de singularidade
$Q$	Parâmetro de entrada da função de enriquecimento de singularidade
$\theta$	Ângulo entre as direções $x$ e de crescimento da trinca
$\sigma_{yy}$	Tensão normal na direção $y$

$\kappa_0$	Valor da deformação equivalente a partir do qual o processo de dano se inicia
$f_e$	Limite de resistência equivalente
$\tilde{E}$	Módulo de elasticidade equivalente
$E_0$	Módulo de elasticidade elástico
$E_2$	Módulo tangente do ramo descendente da lei bilinear
$f_c$	Tensão relativa ao limite de resistência à compressão
$f_t$	Tensão relativa ao limite de resistência à tração
$h$	Comprimento característico
$G_f$	Energia de fratura, definida como a energia necessária para criar uma unidade de área de uma fissura contínua
$\beta_r$	Fator de retenção ao cisalhamento
$\varepsilon_c$	Deformação relativa ao limite elástico na compressão
$\sigma_{xx}$	Tensão normal na direção $x$
$D_z$	Deslocamento na direção $z$

## Capítulo 5

$\varpi_j$	Nuvem de elementos
$N_j(\mathbf{x})$	Função de forma que compõe a PU
$\mathcal{I}_j$	Conjunto de funções de enriquecimento
$L_{ji}(\mathbf{x})$	Função de enriquecimento, também referida como de aproximação local
$\{\phi_{ji}\}_{i=1}^q$	Função de forma do MEFG



$\tilde{\mathbf{u}}(\mathbf{x})$	Aproximação de deslocamento do MEFM
$\mathbf{u}_j$	Parâmetro nodal convencional, associado à $N_j(\mathbf{x})$
$\{\mathbf{b}_{ji}\}_{i=2}^q$	Parâmetros nodais adicionais, associados às $\{N_j(\mathbf{x}) \cdot L_{ji}(\mathbf{x})\}_{i=2}^q$
$H(\xi)$	Função degrau clássica de <i>Heaviside</i> , também referida como descontínua
$\xi$	Posição em relação à descontinuidade assumida em $\xi = 0$
$\mathbf{b}_k$	Parâmetro nodal adicional, associado à $N_k(\mathbf{x})H(\xi)$
$k$	Índice dos nós enriquecidos com função degrau

## Capítulo 6

$\mathbf{x}_d$	Ponto da superfície da descontinuidade
$S_d$	Superfície da descontinuidade
$V^+$	Subdomínio “positivo” do corpo dividido pela descontinuidade
$V^-$	Subdomínio “negativo” do corpo dividido pela descontinuidade
$\mathbf{n}$	Vetor normal unitário da superfície da descontinuidade
$\dot{\boldsymbol{\sigma}}^+$	Taxa de tensão no lado “positivo” da superfície da descontinuidade
$\dot{\boldsymbol{\sigma}}^-$	Taxa de tensão no lado “negativo” da superfície da descontinuidade
$[[\dot{\boldsymbol{\sigma}}]]$	Salto na taxa de tensão
$[[\dot{\boldsymbol{\epsilon}}]]$	Salto na taxa de deformação
$u_1$	Componente do deslocamento na direção 1
$\dot{u}_1$	Taxa do componente do deslocamento na direção 1
$c$	Multiplicador escalar arbitrário

$\dot{\mathbf{u}}$	Taxa de deslocamento
$\mathbf{c}$	Tensor de primeira ordem arbitrário
$\dot{\boldsymbol{\epsilon}}^+$	Taxa de deformação no lado “positivo” da superfície da descontinuidade
$\dot{\boldsymbol{\epsilon}}^-$	Taxa de deformação no lado “negativo” da superfície da descontinuidade
$\mathbf{D}$	Tensor constitutivo tangente de quarta ordem, também referido como de rigidez tangente
$\mathbf{D}^+$	Tensor de rigidez tangente no lado “positivo” da superfície da descontinuidade
$\mathbf{D}^-$	Tensor de rigidez tangente no lado “negativo” da superfície da descontinuidade
$\dot{\epsilon}$	Magnitude do termo salto na condição de <i>continuidade de deslocamento</i>
$\mathbf{m}$	Tensor de primeira ordem unitário denominado <i>vetor de polarização</i>
$\mathbf{Q}$	Tensor de segunda ordem denominado <i>tensor de localização</i> , também referido como <i>tensor acústico</i>
$\mathbf{D}_e$	Tensor de rigidez elástico
$\mathbf{Q}_e$	<i>Tensor acústico</i> correspondente ao tensor de rigidez elástico
$Q_{ik}$	Componente $ik$ do <i>tensor acústico</i>
$D_{ijkl}$	Componente $ijkl$ do tensor de rigidez tangente
$n_j$	Componente $j$ do vetor normal unitário da superfície da descontinuidade
$n_l$	Componente $l$ do vetor normal unitário da superfície da descontinuidade
$\psi$	Ângulo formado entre as direções $x$ e do vetor normal unitário da superfície da descontinuidade

$\mathbf{D}^S$  Tensor constitutivo secante de quarta ordem, também referido como de rigidez secante

$\boldsymbol{\varepsilon}$  Tensor de deformações

## Capítulo 7

$\Omega$  Corpo cortado por uma descontinuidade  $\Gamma_d$

$\Gamma_d$  Descontinuidade no corpo  $\Omega$

$\mathbf{u}$  Campo de deslocamentos

$\Omega^+$  Subdomínio “positivo” do corpo dividido pela descontinuidade  $\Gamma_d$

$\Omega^-$  Subdomínio “negativo” do corpo dividido pela descontinuidade  $\Gamma_d$

$\hat{\mathbf{u}}$  Parcela que compõe a parte contínua de  $\mathbf{u}$

$[[\mathbf{u}]]$  Parcela que compõe a parte descontínua de  $\mathbf{u}$

$H$  Função de *Heaviside*

$[[\mathbf{u}]]_{\mathbf{x} \in \Gamma_d}$  Salto de deslocamentos na descontinuidade  $\Gamma_d$

$\boldsymbol{\varepsilon}$  Campo de deformações

$\delta_{\Gamma_d}$  Função delta de *Dirac* centralizada na descontinuidade  $\Gamma_d$

$\mathbf{n}$  Vetor normal unitário da superfície da descontinuidade; Matriz dos componentes deste vetor;

$()^s$  Indica que somente a parte simétrica dos tensores está sendo considerada

$\delta(x)$  Função generalizada em  $x$

$\mathbf{N}$  Matriz das funções de forma convencionais de MEF

$\mathbf{a}$  Vetor dos graus de liberdade nodal convencionais

$\mathbf{b}$	Vetor dos graus de liberdade nodal adicionais
$\mathbf{B}$	Matriz de aproximação das deformações
$n_x$	Componente do vetor normal unitário $\mathbf{n}$ na direção $x$
$n_y$	Componente do vetor normal unitário $\mathbf{n}$ na direção $y$
$n_z$	Componente do vetor normal unitário $\mathbf{n}$ na direção $z$
$\boldsymbol{\eta}$	Variações admissíveis de deslocamentos
$\boldsymbol{\sigma}$	Campo de tensões
$\bar{\mathbf{t}}$	Forças relacionadas às tensões externas aplicadas no contorno $\Gamma_u$
$\Gamma_u$	Contorno externo do corpo $\Omega$
$\hat{\boldsymbol{\eta}}$	Parcela que compõe a parte contínua de $\boldsymbol{\eta}$
$[[\boldsymbol{\eta}]]$	Parcela que compõe a parte descontínua de $\boldsymbol{\eta}$
$[[\boldsymbol{\eta}]]_{\Gamma_d}$	Variação do salto de deslocamentos na descontinuidade $\Gamma_d$
$\mathbf{t}$	Tensão coesiva agindo na descontinuidade $\Gamma_d$
$\mathbf{a}'$	Variações dos graus de liberdade nodal $\mathbf{a}$
$\mathbf{b}'$	Variações dos graus de liberdade nodal $\mathbf{b}$
$\mathbf{f}_a^{\text{int}}$	Vetor de forças equivalentes às tensões internas correspondentes à $\mathbf{a}'$
$\mathbf{f}_b^{\text{int}}$	Vetor de forças equivalentes às tensões internas correspondentes à $\mathbf{b}'$
$\dot{\boldsymbol{\sigma}}$	Taxa de tensão
$\mathbf{D}$	Matriz constitutiva tangente, também referida como de rigidez tangente
$\dot{\boldsymbol{\epsilon}}$	Taxa de deformação

$\dot{\mathbf{a}}$	Velocidades dos deslocamentos nodais convencionais
$\dot{\mathbf{b}}$	Velocidades dos deslocamentos nodais adicionais
$\dot{\mathbf{t}}$	Taxa de tensão coesiva
$\mathbf{T}$	Matriz de rigidez tangente da relação entre tensão coesiva e deslocamento na descontinuidade
$[[\dot{\mathbf{u}}]]_{\mathbf{x} \in \Gamma_d}$	Taxa do salto de deslocamentos na descontinuidade $\Gamma_d$
$\mathbf{K}$	Matriz de rigidez do elemento
$d\mathbf{a}$	Deslocamentos incrementais convencionais
$d\mathbf{b}$	Deslocamentos incrementais adicionais
$\mathbf{f}_a^{\text{ext}}$	Vetor de forças equivalentes às forças externas correspondentes à $\mathbf{a}'$
$\mathbf{f}_b^{\text{ext}}$	Vetor de forças equivalentes às forças externas correspondentes à $\mathbf{b}'$
$\mathbf{s}$	Vetor na direção paralela à fissura
$t_n$	Componente normal da tensão coesiva
$t_s$	Componente tangencial da tensão coesiva
$f$	Função de carregamento
$[[u]]_n$	Separação normal entre as superfícies da fissura
$\kappa$	Parâmetro histórico, dado pelo maior valor de $[[u]]_n$ alcançado na análise
$f_t$	Resistência à tração do material
$G_f$	Energia de fratura, definida como a energia necessária para criar uma unidade de área de uma fissura contínua
$d_{\text{init}}$	Rigidez inicial da fissura ao cisalhamento, definida para $\kappa = 0$

$h_s$	Parâmetro relacional de rigidezes da fissura ao cisalhamento
$[[u]]_s$	Deslizamento tangencial relativo das superfícies da fissura
$d_{\kappa=1,0}$	Rigidez da fissura ao cisalhamento, definida para $\kappa = 1, 0$
$\dot{t}_n$	Taxa do componente normal da tensão coesiva
$\dot{t}_s$	Taxa do componente tangencial da tensão coesiva
$[[\dot{u}]]_n$	Taxa da separação normal entre as superfícies da fissura
$[[\dot{u}]]_s$	Taxa do deslizamento tangencial relativo das superfícies da fissura
<b>S</b>	Matriz de rigidez secante da relação entre tensão coesiva e deslocamento na descontinuidade

## Capítulo 8

$w$	Valor da função de peso Gaussiana em um ponto de Gauss
$l$	Parâmetro que determina quão rapidamente a função de peso diminui longe do ponto em análise
$r$	Distância de um ponto de Gauss ao ponto em análise
$l_c$	Comprimento interno do elemento
$A_e$	Área média de um elemento finito
$f_t$	Resistência à tração
$G_f$	Energia de fratura

## Capítulo 9

$E$	Módulo de elasticidade
$\nu$	Coefficiente de Poisson

$f_t$	Resistência à tração
$G_f$	Energia de fratura
$d_{init}$	Rigidez inicial ao cisalhamento
$D_y$	Deslocamento na direção $y$
$\sigma_{yy}$	Tensão normal na direção $y$
$E_0$	Módulo de elasticidade elástico
$f_c$	Resistência à compressão
$\varepsilon_c$	Deformação relativa ao limite elástico na compressão
$h$	Comprimento característico
$\beta_r$	Fator de retenção ao cisalhamento
$\sigma_{xx}$	Tensão normal na direção $x$
$f_e$	Limite de resistência equivalente
$\kappa_0$	Valor da deformação equivalente a partir do qual o processo de dano se inicia
$\tilde{E}$	Módulo de elasticidade equivalente
$\varepsilon_1$	Primeiro componente de deformação principal
$\tau_{xy}$	Tensão de cisalhamento no plano normal a $x$ na direção $y$

## Apêndice A

$[K]_{j-1}^i$	Matriz de rigidez tangente da iteração $j - 1$ do passo $i$
$\{\delta U\}_j^i$	Vetor de deslocamentos incrementais da iteração $j$ do passo $i$

$\delta\lambda_j^i$	Incremento do fator de carga da iteração $j$ do passo $i$
$\{P\}$	Vetor de cargas de referência
$\{Q\}_{j-1}^i$	Vetor de forças residuais da iteração $j - 1$ do passo $i$
$\{\delta U\}_j^P$	Vetor de deslocamentos incrementais da iteração $j$ associado à carga de referência
$\{\delta U\}_j^Q$	Vetor de deslocamentos incrementais da iteração $j$ associado à carga residual
$\lambda_j$	Fator de carga proporcional da iteração $j$
$\{U\}_j$	Vetor de deslocamentos da iteração $j$
$\{F\}_j$	Vetor de forças equivalentes às tensões internas ao final da iteração $j$
$k$	Componente de deslocamento como parâmetro de controle
$\delta U_j^k$	Incremento de deslocamento para o componente $k$ na iteração $j$
$\delta U_j^{Q^k}$	Incremento de deslocamento para o componente $k$ associado à carga residual na iteração $j$
$\delta U_j^{P^k}$	Incremento de deslocamento para o componente $k$ associado à carga de referência na iteração $j$

## Apêndice C

$p$	Expoente do componente $x$ do monômio $x^p y^q$
$q$	Expoente do componente $y$ do monômio $x^p y^q$
$x^p y^q$	Monômio genérico da função de enriquecimento polinomial



# Resumo

O objetivo desta dissertação de mestrado é a implementação computacional de um sistema gráfico interativo que integra pré-processador, modelos constitutivos, solucionadores de equações não lineares e pós-processador para análise de nucleação e propagação de trincas em meios bidimensionais parcialmente frágeis. A análise fisicamente não linear se inicia empregando-se o Método dos Elementos Finitos Padrão (MEF), através do qual a degradação inicial do meio é simulada de forma distribuída por meio de métodos constitutivos baseados em degradação elástica. O estágio limiar de nucleação de fissuras é indicado pela singularidade do tensor acústico, característica que fornece a condição clássica de localização de deformações. A presença de trincas é simulada através de método cinemático que incorpora as descontinuidades por meio de interpolações enriquecidas, com base no Método dos Elementos Finitos Generalizados (MEFG), ao passo que a propagação destas trincas é também indicada pela singularidade do tensor acústico. A relação entre tensões e deslocamentos no caminho da trinca é baseada em forças coesivas atuantes no plano da trinca. Esta combinação de métodos constitutivo e cinemático apresenta a vantagem da não necessidade de definição prévia do caminho da trinca ou de redefinição da malha durante o processamento. Ao final da análise, o fenômeno de nucleação e os efeitos da presença de trincas, bem como suas propagações, podem ser visualizados utilizando os recursos do pós-processador. As implementações foram realizadas na plataforma INSANE (*Interactive Structural ANalysis Environment*).

Palavras-Chave: Sistema Gráfico Interativo, Nucleação e Propagação de Trincas, Métodos Constitutivo e Cinemático, Método dos Elementos Finitos Generalizados

# Abstract

This master's thesis is concerned about computational implementation of an interactive graphical software that integrates pre-processor, constitutive models, non-linear equation solvers and post-processor for analysis of nucleation and propagation of cracks in two-dimensional domain in quasi-brittle medium. The physically non-linear analysis begins using the Standard Finite Element Method (FEM), through which the initial medium degradation is simulated in a distributed manner employing constitutive methods based on elastic degradation. The threshold stage of cracks nucleation is indicated by singularity of the acoustic tensor, that brings us to the classical strain localization condition. The presence of cracks is simulated through kinematic method that incorporates the discontinuities by using enriched interpolations based on the Generalized Finite Element Method (GFEM), whereas the cracks propagation is also indicated by singularity of the acoustic tensor. The relationship between stresses and displacements in the crack path is based on cohesive forces acting on the crack plane. This combination of constitutive and kinematic methods has the advantage of not having to pre-set crack path or redefine the mesh during processing. At the end of the analysis, the nucleation phenomenon and the effects of the presence of cracks and their propagation can be viewed using the post-processor module tools. The implementations were performed on the INSANE (*INteractive Structural ANalysis Environment*) platform.

Keywords: Interactive Graphical Software, Nucleation and Propagation of Cracks, Constitutive and Kinematic Methods, Generalized Finite Element Method

# Agradecimentos

À minha família, em especial à minha mãe, pelo esforço e sacrifício para minha criação e dos meus irmãos.

À minha namorada, pelo amor, companheirismo e paciência nas inúmeras horas ausentes, dedicadas ao desenvolvimento deste trabalho.

Ao professor Roque, pela confiança depositada em mim, pela amizade, pelo entusiasmo, pela disponibilidade e pela excelência na orientação deste trabalho.

Ao professor Samuel, pela atenção, pelo interesse e pela excelência na coorientação deste trabalho, mesmo antes de o oficializarmos como coorientador.

Aos demais professores do Departamento de Engenharia de Estruturas da UFMG, pela contribuição à minha formação e no desenvolvimento deste trabalho.

Aos funcionários do Departamento de Engenharia de Estruturas da UFMG, pela presteza e qualidade nos serviços.

À PUC-MG, em especial aos meus eternos professores, pela sólida formação que me proporcionou na graduação.

À UFMG, por possibilitar a consolidação da minha formação através do Programa de Pós-Graduação em Engenharia de Estruturas.

Aos meus amigos do mestrado e à equipe INSANE, por compartilharem desta caminhada sempre dispostos a ajudar.

Ao CNPq, pelo apoio financeiro.

# Capítulo 1

## INTRODUÇÃO

A Análise Estrutural é a parte da Mecânica que estuda o comportamento de estruturas quando solicitadas por agentes externos. Uma maneira de aumentar a acurácia de predição deste comportamento é empregar métodos que permitam a adaptação da análise em função do comportamento mecânico corrente do meio material, possibilitando, por exemplo, a consideração do fenômeno de plastificação em meios elastoplásticos ou da fissuração em meios parcialmente frágeis.

Conforme Karihaloo (1995), os materiais parcialmente frágeis são assim classificados por exibirem na relação tensão-deformação um endurecimento moderado antes do alcance da resistência à tração seguido de um ramo de amolecimento, caracterizado pelo decréscimo gradual das tensões. No caso do concreto, este ramo é devido aos mecanismos de tenacidade na zona de processo de fratura, que, segundo Shah et al. (1995), referem-se à presença de microfissuras e aos empecilhos à propagação das fissuras.

Conforme Süsserkind (1981), as estruturas se compõem de uma ou mais peças, ligadas entre si e ao meio exterior de modo a formar um conjunto estável, isto é, um conjunto capaz de receber solicitações externas, absorvê-las internamente e transmiti-las até seus apoios, onde estas solicitações externas encontrarão seu sistema estático equilibrante. As peças que compõem as estruturas possuem, evidentemente, três dimensões, porém, quanto a classificação espacial da análise, três casos podem ocorrer:

- duas dimensões são desprezáveis em relação à terceira e degeneradas, sendo a análise unidimensional;
- uma dimensão é desprezável em relação às outras duas e degenerada, sendo a análise bidimensional;
- as três dimensões são consideráveis, sendo a análise tridimensional.

Vários são os modelos distribuídos e discretos para tratamento de dano em materiais parcialmente frágeis propostos para análise bidimensional. Desta forma, visando um sistema capaz de simular o comportamento de estruturas de concreto considerando as diferentes fases de degradação, propôs-se o desenvolvimento de um sistema gráfico interativo para análise de nucleação e propagação de trincas em meios bidimensionais parcialmente frágeis. Considerando os conceitos das mecânicas do dano e da fratura, tal análise é processada em três estágios, a saber: degradação distribuída, nucleação de fissuras e propagação de trincas.

O fenômeno de degradação distribuída é analisado por meio de métodos constitutivos. Estes métodos consideram que o meio, apesar de degradado, permanece contínuo, mas que as propriedades do material em estudo se modificam nas regiões submetidas a esforços superiores a determinados parâmetros de resistência. Esta abordagem permite modelar a deterioração sem a consideração da existência de trincas discretas. A continuação do processo de análise, no entanto, pode implicar numa degradação cuja intensidade justifique, visando tornar a análise mais realista, a introdução de descontinuidades na geometria (formação de trincas discretas).

Este estágio limiar é dito nucleação de fissuras e é avaliado através de uma medida de nucleação. Por ser desejável que tal medida seja comum a qualquer método constitutivo, ou seja, que esta medida seja independente de peculiaridades de um método constitutivo particular, definiu-se como medida de nucleação a singularidade do tensor acústico, uma medida que envolve as grandezas do método constitutivo adotado e que caracteriza a condição clássica de localização de deformações.

As descontinuidades são consideradas no decorrer da análise através do Método dos Elementos Finitos Generalizados (MEFG) (Babuska et al., 1994; Duarte e Oden, 1995, 1996), um método numérico de representação não-geométrica cinemático que permite o enriquecimento das funções de Partição da Unidade (PU) de tal maneira que as descontinuidades passam a ser incorporadas nas interpolações, embutindo-se nos elementos os efeitos da presença das trincas nos campos de deslocamentos, de deformações e de tensões.

As forças de coesão atuantes nos planos das trincas são simuladas incorporando-se ao processo de enriquecimento um modelo constitutivo discreto. Consiste na relação entre tensões e deslocamentos no caminho da trinca e baseia-se no conceito de fissura coesiva (ou fictícia), em particular no modelo de Hillerborg et al. (1976), no qual todas as deformações inelásticas, que ocorrem na zona de processo de fratura, são representadas em uma linha através das forças coesivas que agem na trinca, ou em uma extensão fictícia desta.

Uma análise com as características descritas é dita análise fisicamente não linear, com a qual é possível simular o comportamento da estrutura de forma mais realista, porém, tal análise só é possível com o uso de recursos computacionais adequados.

O sistema computacional INSANE (*INteractive Structural ANalysis Environ-ment*) é um projeto de software livre, implementado em linguagem Java segundo o paradigma de Programação Orientada a Objetos, desenvolvido no Departamento de Engenharia de Estruturas da Escola de Engenharia da Universidade Federal de Minas Gerais e disponível em <http://www.insane.dees.ufmg.br>. Este sistema dispõe de diversos recursos para análise fisicamente não linear e pôde ser aperfeiçoado para a análise proposta, por ser um ambiente computacional segmentado, amigável a mudanças e escalável em complexidade. Para tal aperfeiçoamento, foram necessárias adaptações no núcleo numérico do sistema, porém, para uma interação mais amigável com o usuário, também foi desenvolvido um pré-processador gráfico interativo com recursos condizentes com a análise proposta.

## 1.1 Objetivos

### 1.1.1 Objetivo Geral

Esta dissertação tem por objetivo apresentar a implementação computacional de um sistema gráfico interativo para modelagem de nucleação e propagação de trincas em meios bidimensionais parcialmente frágeis, cuja análise é processada em três estágios: degradação distribuída, nucleação de fissuras e propagação de trincas.

### 1.1.2 Objetivos Específicos

Na Mecânica do Dano, a resistência de uma estrutura carregada é determinada em função da evolução de um campo de defeitos continuamente distribuídos, enquanto na Mecânica da Fratura, a resistência é determinada em função da evolução de defeitos particulares (Guello, 2002).

Assim, do ponto de vista conceitual, o tema desta dissertação se insere no campo da Mecânica do Dano, quando da degradação pré-nucleação, bem como no campo da Mecânica da Fratura, quando da consideração de descontinuidades e de suas propagações.

Do ponto de vista numérico, o tema envolve o Método dos Elementos Finitos (MEF) e sua generalização (MEFG), bem como os recursos necessários à solução do problema fisicamente não linear: Modelos Constitutivos Distribuídos, Critérios de Nucleação, Modelos Constitutivos Discretos e Métodos de Solução das Equações Não Lineares.

Do ponto de vista computacional, a dissertação trata de Sistemas Orientados a Objetos, Estruturas de Dados para Subdivisão Planar, Técnicas para Geração de Malhas e Interfaces Gráficas Interativas.

Para combinar o arcabouço conceitual das Mecânicas do Dano e da Fratura com os recursos numéricos e computacionais acima listados, visando ao objetivo geral, os seguintes objetivos específicos foram perseguidos:

1. Aperfeiçoamento da aplicação gráfica interativa de entrada de dados para análise bidimensional do sistema INSANE;
2. Integração do pré-processador, processador e pós-processador, com recursos necessários para simulação de problemas fisicamente não lineares;
3. Avaliação do processo de degradação do material de forma distribuída, empregando-se métodos constitutivos;
4. Definição de uma medida de nucleação capaz de determinar o estágio limiar de introdução das descontinuidades no processo;
5. Incorporação das descontinuidades nas interpolações enriquecidas dos elementos finitos que as contêm, empregando-se o MEEG;
6. Introdução de mecanismos de tenacidade na zona de processo de fratura, aqueles atuantes nas superfícies da trinca formada, através de forças coesivas.

## 1.2 Organização do Texto

Esta dissertação está organizada em 10 capítulos e 4 apêndices.

No capítulo 1, apresenta-se a Introdução do trabalho, justificando-se a escolha do tema, além de definir os objetivos e mostrar a organização do texto.

No capítulo 2, os recursos empregados nas Aplicações Gráficas Interativas do Sistema INSANE são abordados, destacando-se Sistemas Orientados a Objetos, Padrões de Projeto de Software, Estrutura de Dados para Subdivisão Planar e Técnicas de Geração de Malhas.

No capítulo 3, a Metodologia de Integração do Sistema Gráfico é apresentada, com enfoque na classe *Combinable*, responsável por intermediar a comunicação da aplicação gráfica de entrada de dados com o núcleo numérico do sistema.



No capítulo 4, Exemplos de Modelagem no Sistema Gráfico Integrado são apresentados, destacando-se o emprego deste sistema para os seguintes tipos de análise bidimensional estática:

- Linear com carregamento no plano pelo MEF e pelo MEFG;
- Linear com carregamento normal ao plano (placas) pelo MEF;
- Fisicamente não linear com carregamento no plano pelo MEF e pelo MEFG.

No capítulo 5, apresenta-se uma classificação dos principais Métodos Numéricos para Fissuração, proposta por Ingraffea e Wawrzynek (2004), com enfoque nos Métodos Constitutivos e no Método dos Elementos Finitos Generalizados - Um Método Cinemático, cuja combinação permite a modelagem da degradação distribuída evoluindo para formação de trincas discretas sem a necessidade de definição prévia do caminho da trinca ou de redefinição da malha durante o processamento.

No capítulo 6, apresenta-se como medida de Nucleação de Fissuras a singularidade do Tensor Acústico, definida pelo seu embasamento físico e por ser aplicável independentemente do modelo constitutivo distribuído adotado.

No capítulo 7, um Modelo Constitutivo Discreto, responsável por simular as forças de coesão atuantes nos planos das trincas, é formulado para uma lei coesiva particular, porém, sem perda da generalidade do sistema.

No capítulo 8, apresenta-se o Projeto Orientado a Objetos implementado no Núcleo Numérico do INSANE para modelagem de Nucleação e Propagação de Trincas coesivas em meios bidimensionais, com enfoque na classe *DiscontinuityByGFEM*, responsável pela simulação e gerenciamento de trincas via abordagem de enriquecimento nodal. Destacam-se as intervenções no arcabouço computacional de análise pelo MEFG, bem como em outras frentes do sistema, para inserção desse Projeto.

No capítulo 9, apresentam-se Simulações Numéricas para avaliar o desempenho e a generalidade do modelo numérico implementado, realizadas através da opção de a análise fisicamente não linear detectar nucleação de trincas e processar suas propagações, incluída no sistema gráfico integrado.

No capítulo 10, são apresentadas as Considerações Finais e algumas sugestões para trabalhos futuros.

No apêndice A, Processos Incrementais-Iterativos, alguns dos recursos do sistema INSANE para solução de problemas não lineares, são abordados.

No apêndice B, para melhor compreensão das tarefas atribuídas à Classe *Combinable*, apresentam-se suas principais variáveis e seus principais métodos.

No apêndice C, apresenta-se o Sistema Gráfico Integrado, destacando-se a organização do pré-processador e sua integração ao processador e ao pós-processador, além de expor os recursos disponibilizados no sistema.

No apêndice D, para melhor compreensão das tarefas atribuídas à Classe *DiscontinuityByGFEM*, apresentam-se suas principais variáveis e seus principais métodos.

## Capítulo 2

# APLICAÇÕES GRÁFICAS INTERATIVAS DO INSANE

Este capítulo apresenta alguns dos recursos necessários para implementação do pré-processador gráfico interativo no sistema INSANE, discutindo o Paradigma de Programação Orientada a Objetos, Padrões de Projeto de Software, Estruturas de Dados para Subdivisão Planar e Métodos para Geração de Malhas, apresentando suas aplicações neste ambiente.

### 2.1 Sistemas Orientados a Objetos

O processo de desenvolvimento de software orientado a objetos, empregado no sistema INSANE, compreende três fases principais, conforme Gonçalves (2004). Inicialmente, na fase de análise, procura-se enfatizar a descoberta e descrição dos objetos - ou conceitos - do domínio do problema. Em seguida, na fase de projeto, procura-se definir os elementos lógicos de software, seus atributos e métodos. Finalmente, durante a fase de construção, os componentes do projeto são implementados em uma linguagem de programação que suporte o paradigma da programação orientada a objetos.

O INSANE usa a linguagem de programação JAVA, diversas APIs (*Application Program Interface*) e pacotes gráficos disponíveis para esta linguagem.

Conceitualmente, o paradigma de programação orientada a objetos possui características que o define, destacando-se:

### **2.1.1 Classes e Objetos**

Uma classe é um componente de programa que descreve a “estrutura” e o “comportamento” de um grupo de objetos semelhantes - isto é, as informações que caracterizam o estado desses objetos e as ações (ou operações) que eles podem realizar. Os objetos de uma classe - também chamados de instâncias da classe - são criados durante a execução de programas. Uma classe é formada, essencialmente, por construtores de objetos dessa classe, variáveis e métodos. A criação de um objeto dessa classe consiste na criação de cada uma das variáveis do objeto, especificadas na classe. Os valores armazenados nessas variáveis determinam o estado do objeto. Uma variável de um objeto é também chamada de “atributo” desse objeto e pode ser modificada durante a execução do programa (Camarão, 2003).

### **2.1.2 Coleções de Objetos**

Muitas aplicações para as quais desejam-se desenvolver um programa, consistem em sistemas bastante complexos. Uma maneira natural de lidar com a complexidade de um sistema é dividi-lo em subsistemas mais simples, de maneira que o comportamento do sistema, como um todo, possa ser expresso em termos dos comportamentos de seus subsistemas e das interações entre eles. Linguagens de programação mais modernas oferecem recursos para construção de um programa como uma coleção de componentes de programa, com interfaces bem definidas, que especificam as interações entre esses componentes. Objetos constituem limites naturais para construções de abstrações de dados: todas as informações referentes a uma dada entidade são confinadas em um determinado objeto, que se relaciona com outros objetos mediante uma interface bem definida (Camarão, 2003).

### 2.1.3 Abstração

Abstrair significa decompor um sistema complicado em suas partes fundamentais e descrevê-las em uma linguagem simples e precisa. A descrição das partes de um sistema implica atribuir-lhes um nome e descrever suas funcionalidades. Consiste na não necessidade de se ter ciência da maneira com que uma dada operação é realizada devido à clareza na descrição do comando que aciona tal operação (Goodrich e Tamassia, 2007).

### 2.1.4 Encapsulamento

Este princípio estabelece que os diferentes componentes de um sistema de software não devem revelar detalhes internos de suas respectivas implementações. Genericamente, o princípio do encapsulamento propõe que todos os componentes de um grande sistema de software operem dentro de uma filosofia de conhecer o mínimo necessário sobre os demais. Uma das maiores vantagens do encapsulamento é que ele oferece ao programador liberdade na implementação dos detalhes do sistema. A única restrição ao programador é manter a interface abstrata que é percebida pelos demais componentes (Goodrich e Tamassia, 2007).

### 2.1.5 Modularidade

Sistemas modernos de software normalmente estão compostos por vários componentes diferentes que devem interagir corretamente, fazendo com que o sistema como um todo funcione de forma adequada. Para manter essas interações corretas é necessário que os diversos componentes estejam bem organizados. Na abordagem orientada a objetos, essa organização se centra no conceito de modularidade. A modularidade refere-se a uma estrutura de organização na qual os diferentes componentes de um sistema de software são divididos em unidades funcionais separadas (Goodrich e Tamassia, 2007).

### 2.1.6 Herança

Para evitar código redundante, o paradigma de orientação a objetos oferece uma estrutura hierárquica e modular para reutilização de código através de uma técnica conhecida como herança. Esta técnica permite projetar classes genéricas que podem ser especializadas em classes mais particulares, onde as classes especializadas reutilizam o código das mais genéricas. A classe genérica, também conhecida por classe base ou superclasse, define variáveis de instância “genéricas” e métodos que se aplicam em uma variada gama de situações. A classe que especializa, estende ou herda uma superclasse não necessita fornecer uma nova implementação para os métodos genéricos, uma vez que os herda. Deve apenas definir aqueles métodos que são especializados para esta subclasse em particular (também conhecida como classe derivada) (Goodrich e Tamassia, 2007).

### 2.1.7 Polimorfismo

No contexto de projeto orientado a objetos, polimorfismo refere-se à habilidade de uma variável de objeto de assumir formas diferentes, permitindo manipular de forma genérica instâncias (objetos) de classes que herdaram de uma mesma classe ancestral. Linguagens orientadas a objetos referenciam objetos usando variáveis referência. Algumas linguagens orientadas a objetos também oferecem um tipo de polimorfismo “em cascata”, que é mais precisamente conhecido como sobrecarga de métodos. A sobrecarga ocorre quando uma única classe tem vários métodos com o mesmo nome, desde que cada um tenha uma assinatura diferente. A assinatura de um método é uma combinação entre seu nome e o tipo e a quantidade de argumentos que são passados para o mesmo. Dessa forma, mesmo que vários métodos de uma classe tenham o mesmo nome, eles são distinguíveis pelo compilador pelo fato de terem diferentes assinaturas, ou seja, na verdade são desiguais (Goodrich e Tamassia, 2007).

O paradigma de programação orientada a objetos viabiliza expansões e mudanças, sem a necessidade de reescrever o código. Da perspectiva das aplicações gráficas permite, por exemplo, segundo Gonçalves (2004), o encapsulamento de rotinas de execução em objetos, com posterior associação destes objetos aos elementos de interface gráfica com o usuário (botões e menus) e aos dispositivos de entrada (teclado e “mouse”), conforme apresentado em seguida em Padrões de Projeto de Software, ao se discutir sobre o padrão *Command*.

## 2.2 Padrões de Projeto de Software

Segundo Gonçalves (2004), padrões de projeto podem ser definidos como a estrutura básica de um projeto de software bem-sucedido, capaz de fornecer um esquema para os subsistemas ou componentes de um sistema de software a ser projetado. Esse esquema deve ser específico para resolver o problema em questão e suficientemente genérico para atender a futuros problemas e requisitos. A utilização de padrões de projeto propicia evitar ou minimizar o reprojeto.

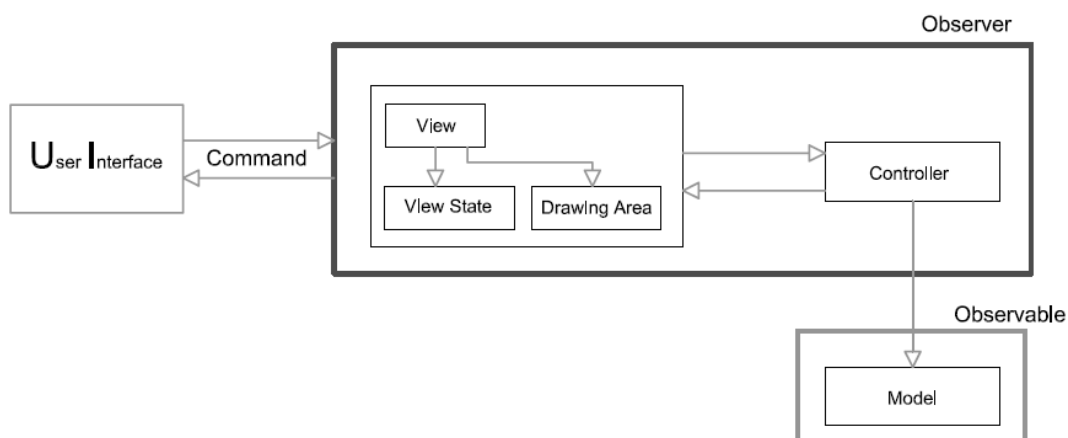
Conforme Penna (2007), o sistema INSANE apresenta uma arquitetura em camadas baseada na combinação de três padrões de projeto de software: o padrão *Model-View-Controller (MVC)*, que permite a independência entre as camadas modelo e vista; o padrão *Observer* que, através de um mecanismo de propagação de mudanças, permite a sincronização do modelo com os diversos pares “vista-controlador”; e o padrão *Command* que encapsula as ações do programa em classes separadas de forma organizada. Esta segmentação favorece a expansão do programa e a criação de novos comandos.

### 2.2.1 Padrão *Model-View-Controller*

Segundo Gonçalves (2004), visando separar o modelo de sua representação, as aplicações gráficas interativas do INSANE são baseadas no padrão de projeto *Model-View-Controller (MVC)*. A utilização desta metáfora de programação permite que o controle da criação do modelo, através de interação com o usuário, e a visualização do mesmo sejam implementados independentemente do modelo adotado, minimizando as tarefas de manutenção e expansão da aplicação. A implementação segundo o padrão *MVC* permite o aperfeiçoamento gradual da aplicação através de mudança de plataforma, criação de diversas vistas sincronizadas com o modelo, substituição ou atualização das diversas vistas e disponibilização “on-line” do sistema.

Existe um ciclo de vida para cada uma das atividades executadas pelo programa. Este ciclo permite que o usuário faça alterações no modelo e visualize o resultado a cada alteração, até que consiga o resultado desejado. O referido ciclo compõe-se de: especificação do usuário, atualização do modelo e visualização.

O padrão de projeto *MVC* pode ser usado para a implementação do ciclo de vida de cada atividade. Este padrão divide a aplicação em três componentes: modelo, vista e controlador. A Figura 2.1 ilustra o padrão.



**Figura 2.1:** Arquitetura do INSANE (Penna, 2007).



O modelo contém todos os dados do sistema, sendo independente das saídas e entradas de dados. A vista apresenta para o usuário as informações armazenadas no modelo. Cada controlador é associado a um componente vista, sendo o responsável pela percepção das entradas do usuário e tradução das mesmas em requisições de serviços para os componentes modelo e vista. Todas as requisições dos usuários devem ser feitas através dos controladores (Buschmann et al., 1996). Conforme será discutido, existe um mecanismo de propagação de mudanças que garante a consistência e a comunicação entre os componentes do padrão *MVC*.

### 2.2.2 Padrão *Command*

Conforme Gonçalves (2004), o uso do padrão *Command* no INSANE permite o encapsulamento de rotinas de execução em objetos, a associação destes objetos a elementos de interface gráfica com o usuário (GUI) e dispositivos de entrada (teclado e “*mouse*”), a execução de uma mesma rotina disparada por diferentes elementos de GUI e possibilita um incremento na modularidade de seu código. O encapsulamento das rotinas de execução possibilita também que a realização de alterações nas mesmas não provoque modificações nas classes existentes.

O padrão *Command* baseia-se em uma classe abstrata de mesmo nome, a qual declara uma interface para execução de operações. Na sua forma mais simples, esta interface inclui uma operação abstrata **execute()**. As subclasses concretas de *Command* especificam um par receptor-ação através do armazenamento do receptor como uma variável de instância e pela implementação de **execute()**, para invocar a solicitação. O receptor tem o conhecimento para poder executar a solicitação.

Este padrão desacopla o objeto que invoca a operação daquele que tem o conhecimento para executá-la. Isto proporciona flexibilidade no projeto da interface de usuário. Uma aplicação pode oferecer tanto uma interface gráfica com menus, quanto com botões para algum recurso seu, simplesmente fazendo com que o menu e o botão compartilhem uma instância da mesma classe que implementa *Command*.

O padrão *Command* possibilita a substituição dinâmica de comandos, o que é muito útil para interfaces gráficas sensíveis ao contexto. Pode-se ainda concatenar comandos para compor comandos maiores, propiciando a redução da complexidade destes. Todos estes recursos são possíveis porque o objeto que emite a solicitação não precisa conhecer a execução da solicitação.

O tempo de vida de um objeto *Command* é independente de sua solicitação original, permitindo armazenar comandos e executar suas rotinas em momentos distintos. A operação **execute()**, de *Command*, pode armazenar estados para que o comando possa reverter seus efeitos. Para suportar a operação desfazer, a interface de *Command* deve acrescentar a operação **undo()**, que reverte os efeitos de uma chamada anterior de **execute()**. Os comandos executados devem ser armazenados em uma lista histórica. O nível ilimitado de desfazer e refazer operações é obtido percorrendo esta lista para trás e para frente, chamando operações **undo()** e **execute()**, respectivamente (Gamma et al., 1995).

A Figura 2.2 apresenta os componentes envolvidos com o padrão *Command* e ilustra o relacionamento entre eles. *Receiver* é o componente que sabe como executar as operações associadas a uma solicitação, qualquer classe pode funcionar como um *Receiver* (exemplos: uma área de desenho, um controlador). A classe *Command* declara uma interface para a execução de uma operação. *ConcreteCommand* implementa o processo **execute()** através da invocação das operações correspondentes no *Receiver* e define uma vinculação entre um objeto *Receiver* e uma ação. Quando os comandos podem ser desfeitos, *ConcreteCommand* armazena estados para desfazer o comando antes de invocar **execute()**. O componente *Invoker* (botão e/ou item de menu) é responsável por disparar o processo **execute()** de seu comando associado (Gamma et al., 1995). O componente *Client* (exemplo: uma interface gráfica) cria um objeto *Concrete-Command*, o associa a um *Invoker* e estabelece o seu receptor (*Receiver*).

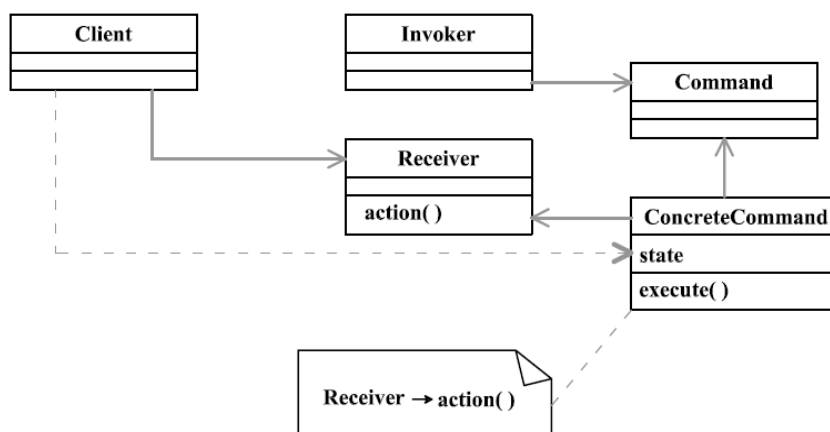


Figura 2.2: Estrutura do padrão *Command* (Gamma et al., 1995).

### 2.2.3 Padrão *Observer*

Segundo Gamma et al. (1995), o padrão *Observer* fornece um *Mecanismo de Propagação de Mudanças* que garante a consistência e a comunicação entre os componentes controlador e vista com o componente modelo, do padrão *MVC*, podendo ser empregado para a atualização das vistas a cada mudança no modelo (Penna, 2007). Este mecanismo efetua seu registro no momento em que um componente controlador ou vista é criado, ligando-o ao modelo do qual ele é dependente. A cada mudança de estado do modelo, o *Mecanismo de Propagação de Mudanças* é disparado, acarretando na execução do procedimento de atualização do componente vista, que exibe ao usuário a informação atualizada. A Figura 2.3a ilustra este padrão.

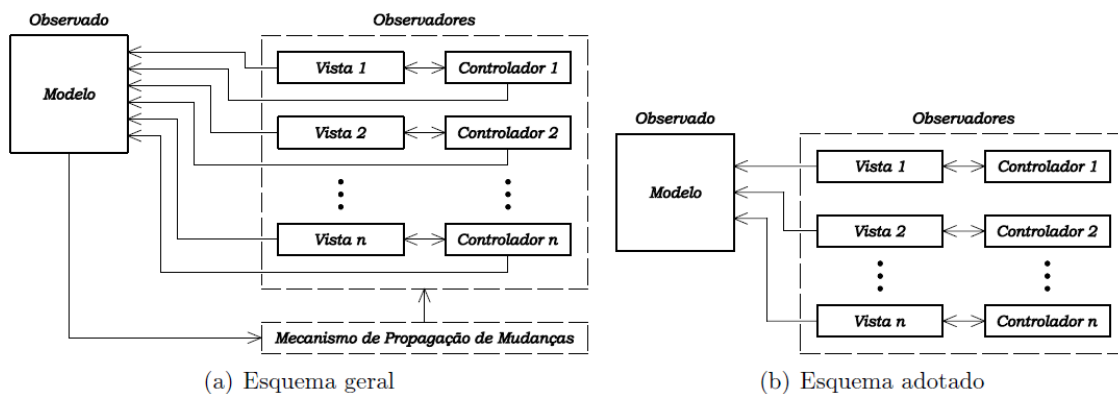


Figura 2.3: Padrão *Observer* (Gonçalves, 2004).

Conforme Gonçalves (2004), o esquema apresentado na Figura 2.3a é mais adequado para o caso geral, quando existem vistas múltiplas que simultaneamente apresentam dados do modelo. Na implementação do pré-processador foram usadas diversas vistas, havendo, porém, apenas uma vista e um controlador ativos em cada atividade. Essa particularidade proporcionou a elaboração de um sistema de funcionamento mais simples, apresentado na Figura 2.3b. Como existe um único controlador ativo e este transmite as requisições de serviços para o modelo, ele mesmo pode ficar encarregado de disparar o processo de atualização da vista, substituindo o *Mecanismo de Propagação de Mudanças* (Figura 2.3a). Cada vista é associada a um único controlador e fornece a este a funcionalidade necessária para manipular a exibição de dados. Listas de desenhos referentes aos dados do modelo são mantidas no controlador ativo, promovendo o isolamento entre o modelo e a vista, uma vez que esta não precisa acessar o modelo para apresentar seus dados, passando a apresentar apenas as listas de desenhos contidas no controlador ativo.

## 2.3 Estrutura de Dados para Subdivisão Planar

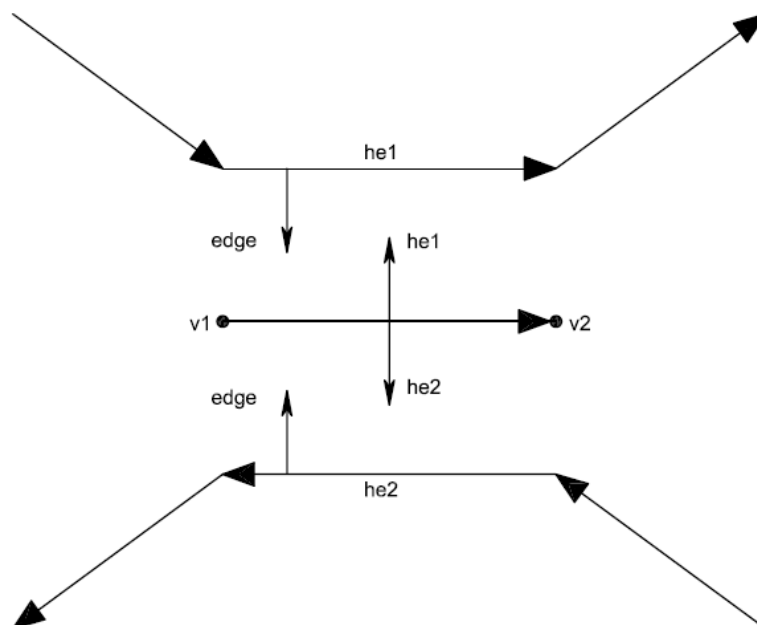
O objeto modelo apresentado na seção 2.2 consiste em uma estrutura de dados responsável por armazenar as entidades topológicas do modelo em análise.

Conforme Penna (2007), o desenvolvimento de modelos geométricos requer o armazenamento de dados de forma estruturada e acessível, tal que possibilite a manipulação de construções e operações geométricas. Tendo isto em vista, são usadas estruturas de dados para subdivisão planar, também chamadas estruturas de adjacências, no tratamento destes modelos. Consiste em uma forma organizada de armazenamento de dados de modelos geométricos computacionais, possibilitando um rápido acesso a estes dados posteriormente.

A estrutura de dados empregada nas aplicações gráficas do sistema INSANE é a estrutura de Semi-Arestas ou “*Half-Edge*”.

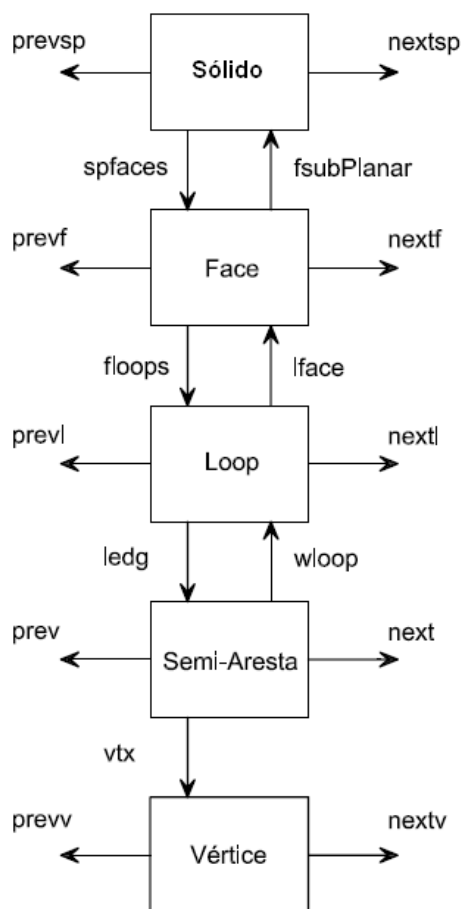
Esta estrutura facilita a pesquisa de entidades  $e$ , de forma simples, soluciona questões fundamentais para a manipulação de dados geométricos como, por exemplo, dado um vértice quais faces são comuns a ele, ou dada uma face quais são seus vértices (Penna, 2007).

A estrutura “*Half-Edge*” permite armazenar informações relativas a uma subdivisão planar, de modo que estas proporcionem a relação de adjacência entre as entidades que compõem o modelo. Esta estrutura é um aprimoramento da estrutura de “Arestas-Aladas”, diferenciando-se pela sua organização topológica. Conforme ilustrado na Figura 2.4, a principal alteração é a transformação de uma aresta (*edge*) em duas semi-arestas, uma à esquerda (*he1*) e uma à direita (*he2*), incidentes nos mesmos vértices ( $v1$  e  $v2$ ), porém, cada semi-aresta armazena um dos vértices como seu início, definindo seu sentido.



**Figura 2.4:** Representação da aresta na estrutura “*Half-Edge*” (Mäntylä, 1987).

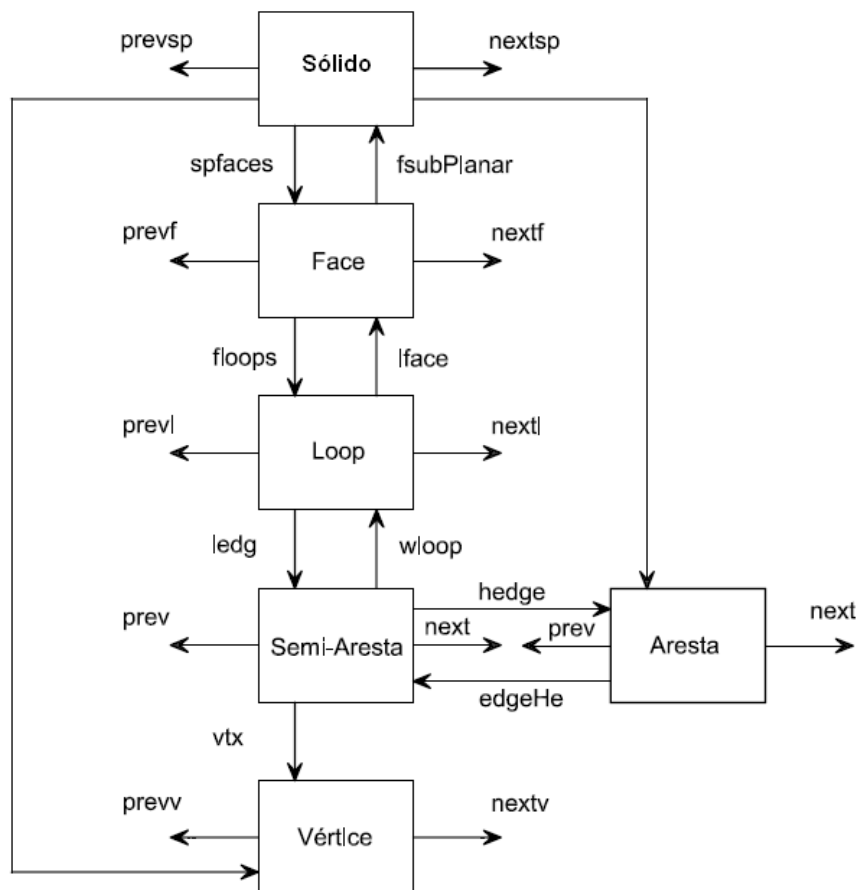
Conforme Mäntylä (1987), esta estrutura de dados se desenvolve em uma hierarquia composta por cinco níveis diferentes, sendo eles o **Sólido**, a **Face**, o **Loop**, a **Semi-Aresta** e o **Vértice**. Cada nível é armazenado em uma lista duplamente encadeada. Esta estrutura hierárquica é ilustrada na Figura 2.5.



**Figura 2.5:** Estrutura “Half-Edge” (Adaptado de Mäntylä (1987)).

No topo da hierarquia tem-se o nível **Sólido**, que possui uma referência ao nível **Face**. Na sequência, tem-se o nível **Face**, que, além de uma referência ao seu respectivo **Sólido**, tem uma referência ao nível **Loop**, uma vez que uma face é composta por um ou mais “*loops*”, sendo estes externos, representando o contorno da face, ou internos, representando furos na face. O nível seguinte é o **Loop**, que possui uma referência à sua respectiva **Face** e, por ser composto por semi-arestas, possui uma referência para o nível **Semi-Aresta**. O nível **Semi-Aresta** possui uma referência para o seu respectivo **Loop** e possui também uma referência para o nível **Vértice**, que é o vértice de início da respectiva semi-aresta. O último nível é o **Vértice**, que não possui referência aos demais níveis.

A estrutura de adjacência como representada até agora não faz conexão entre as faces com a finalidade de compor o modelo geométrico. Para tanto, Del Savio et al. (2004) adicionaram um nível intermediário, conforme ilustrado na Figura 2.6.



**Figura 2.6:** Estrutura “Half-Edge” modificada (Adaptado de Del Savio et al. (2004)).

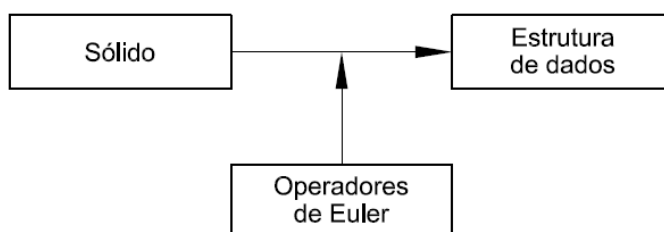
Este nível é a **Aresta**, que possui uma referência para o nível **Semi-Aresta**, uma vez que a aresta é composta por duas semi-arestas, uma à esquerda e uma à direita, incidentes nos mesmos vértices. As arestas também ficam armazenadas em uma lista duplamente encadeada. Com esta alteração, o nível **Sólido** passa a ter também referência para os níveis **Aresta** e **Vértice**, e o nível **Semi-Aresta** uma referência para a sua respectiva **Aresta**.

O armazenamento dos dados de um modelo geométrico na estrutura discutida permite o acesso aos elementos do modelo, por consulta simples, seguindo a ordem da hierarquia (Penna, 2007).

Outros recursos implementados no sistema INSANE relativos à estrutura de dados e fundamentais para o pré-processador gráfico interativo são os operadores de Euler, discutidos a seguir.

### 2.3.1 Operadores de Euler

Segundo Mäntylä (1987), os operadores de Euler criam e manipulam as entidades topológicas (vértices, arestas e faces) que compõem um modelo sólido, para representá-lo numa estrutura de dados, conforme ilustrado na Figura 2.7.



**Figura 2.7:** Atuação dos operadores de Euler (Ferreira, 2008).

Por convenção histórica, os operadores de Euler são normalmente denotados por códigos de nomes abreviados, por exemplo: MEV = *Make Edge, Vertex*. As chaves para esses nomes são usadas conforme apresentado em Mäntylä (1987).

Uma sequência finita de operadores de Euler é capaz de criar uma estrutura de dados segura, de maneira que as relações de conectividade entre as entidades topológicas fiquem consistentes. Para garantir essa consistência, a chamada fórmula de *Euler-Poincaré* (Mäntylä, 1987), apresentada na Equação 2.1, deve ser satisfeita.

$$V - E + F = 2 \cdot (S - H) + R \quad (2.1)$$

onde,

$V$ ,  $E$ ,  $F$ ,  $S$ ,  $H$  e  $R$  são, respectivamente, os números de vértices, arestas, faces, sólidos (volumes independentes), furos passantes (gênero topológico) e “*loops*” internos (*ring*).



Conforme Mäntylä (1987), os operadores de Euler são divididos em duas categorias: os Locais, que alteram a topologia de apenas uma parte do modelo e os Globais, que alteram globalmente a topologia.

Os operadores de Euler, da perspectiva do pré-processador gráfico interativo, são responsáveis pelo desenvolvimento do modelo geométrico e o gerenciamento da estrutura de dados, ou seja, permitem a criação e remoção de entidades topológicas a partir da área interativa do programa, armazenando e gerenciando as vinculações destas entidades em consonância com a estrutura de dados empregada.

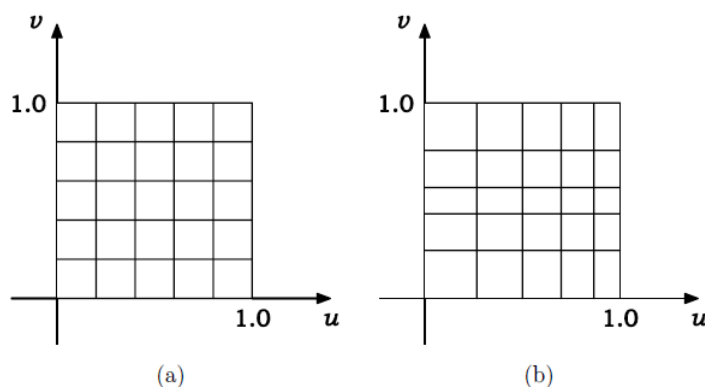
## 2.4 Geração de Malhas

Segundo Fonseca (1989), os métodos de geração de malhas em domínios bidimensionais podem ser classificados em Diretos ou Algébricos e Indiretos ou de Equações Diferenciais.

### 2.4.1 Métodos Diretos ou Algébricos

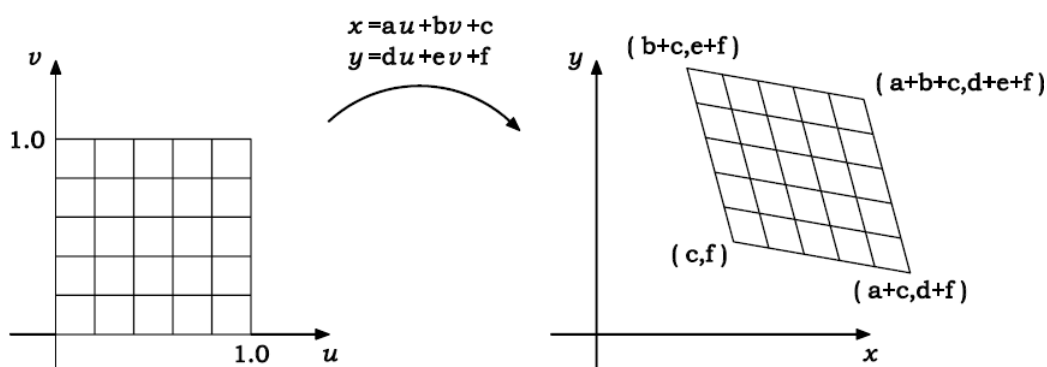
Conforme Gonçalves (2004), os Métodos Diretos ou Algébricos são assim chamados porque geram uma malha sobre o domínio, baseados em algum algoritmo algébrico definido. Estes métodos podem ser subdivididos em Geração de Malhas em Domínios Elementares, Geração de Malhas por Transformação de Coordenadas, Mapeamentos Conformes, Mapeamentos Isoparamétricos, Mapeamentos Transfinitos e Decomposição de Domínio.

Segundo Gonçalves (2004), a Geração de Malhas em Domínios Elementares, Figura 2.8a, não constitui propriamente um método de geração de malhas. Entretanto, as malhas em domínios elementares são úteis para a geração de malhas em domínios mais complexos, usando-se técnicas de transformação de coordenadas ou de mapeamento. No caso de se desejar obter uma maior concentração de pontos numa determinada região, Figura 2.8b, pode-se usar uma função de densidade de malha.



**Figura 2.8:** Malhas em Domínios Elementares (Gonçalves, 2004).

A Geração de Malhas por Transformação de Coordenadas, segundo Gonçalves (2004), é o tipo mais simples de mapeamento e consiste em usar uma transformação de coordenadas que mapeia um domínio elementar no domínio real desejado, conforme ilustrado na Figura 2.9. A desvantagem deste método é que torna-se necessário determinar a transformação de coordenadas que faça o mapeamento do domínio elementar no domínio real, o que nem sempre é simples ou mesmo possível.



**Figura 2.9:** Malhas por Transformação de Coordenadas (Gonçalves, 2004).

O Mapeamento Conforme consiste em associar os pontos dos domínios elementar e real a números complexos. Desta forma, o mapeamento procurado torna-se uma função de variável complexa que mapeia o domínio elementar no domínio real desejado, conforme ilustrado na Figura 2.10. Entre as desvantagens deste método está o fato de que as características da malha são definidas automaticamente, dificultando a produção de uma malha adequada ao problema a ser solucionado.

Este método também exige que se determine a transformação de coordenadas que faça o mapeamento do domínio elementar no domínio real. Além disto, podem surgir dilatações ou adensamentos indesejáveis na malha (Gonçalves, 2004).

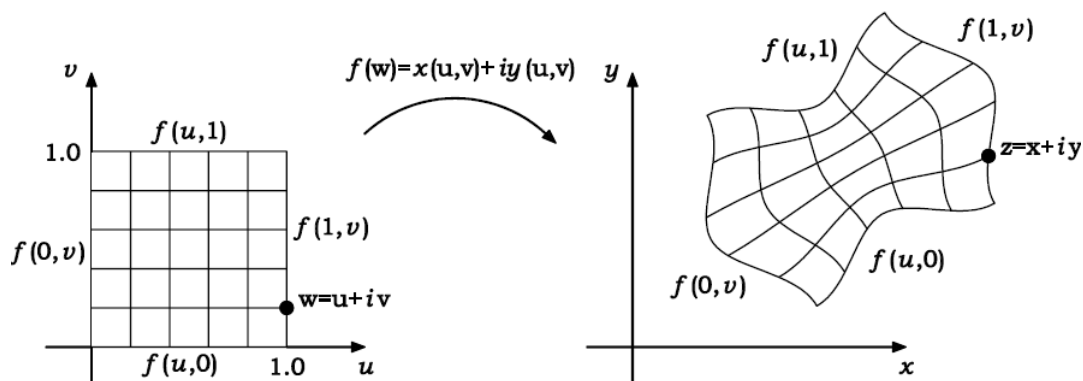


Figura 2.10: Mapeamento Conforme (Gonçalves, 2004).

O Mapeamento Isoparamétrico consiste em obter os valores das coordenadas dos pontos do domínio a partir de valores especificados no contorno, através do uso de funções de interpolação, conforme ilustrado na Figura 2.11. Neste processo, o contorno do domínio não é especificado, mas aproximado por funções de interpolação que passam por pontos especificados. Estas mesmas funções de interpolação são usadas para mapear a malha no domínio. Neste tipo de mapeamento, é possível controlar a densidade de malha através do posicionamento adequado dos nós dos centros dos lados, respeitando certos limites (Gonçalves, 2004).

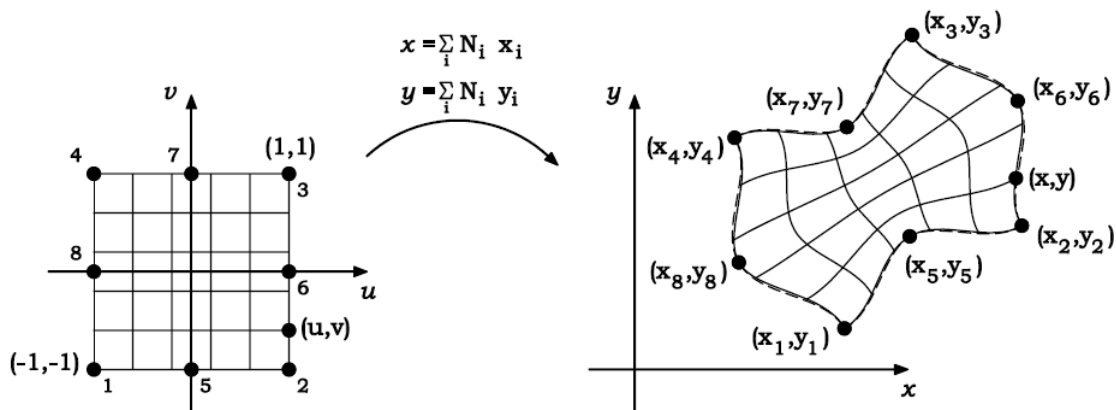
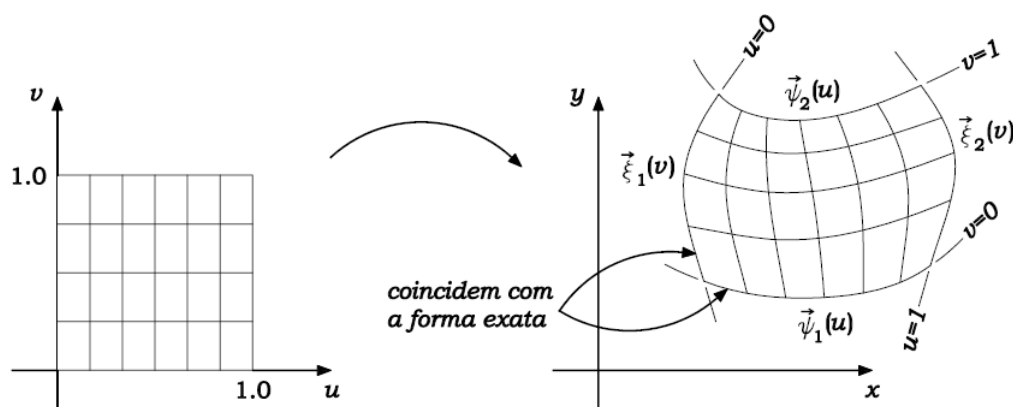


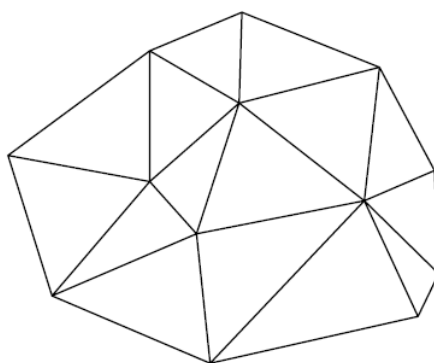
Figura 2.11: Mapeamento Isoparamétrico (Gonçalves, 2004).

Segundo Gonçalves (2004), o Mapeamento Transfinito estabelece sistemas de coordenadas curvilíneas definidos pelo contorno de domínios arbitrários, conforme ilustrado na Figura 2.12. Este método descreve uma superfície aproximada que coincide com a superfície real ou idealizada em um número não enumerável de pontos, propriedade que deu origem ao nome Mapeamento Transfinito. É capaz de modelar o contorno de superfícies sem a introdução de nenhum erro na geometria do mesmo.



**Figura 2.12:** Mapeamento transfinito (Gonçalves, 2004).

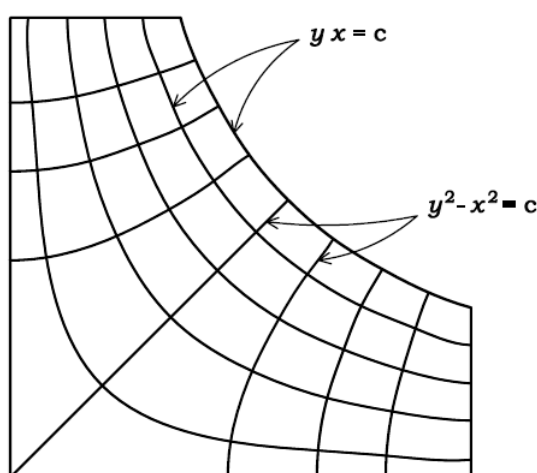
Conforme Gonçalves (2004), o Método de Decomposição de Domínio, ilustrado na Figura 2.13, baseia-se em técnicas de decomposição espacial, como árvore quaternária (*quadtree*), avanço de fronteira e o método de *Delaunay*, sendo uma alternativa relevante nos casos de domínios com formas muito complexas e em casos onde se deseja variar drasticamente a densidade de elementos ao longo do domínio.



**Figura 2.13:** Decomposição de Domínio (Gonçalves, 2004).

## 2.4.2 Métodos Indiretos ou de Equações Diferenciais

Nos Métodos Indiretos ou de Equações Diferenciais, a geração da malha envolve a solução de uma equação diferencial a fim de obter um sistema de coordenadas curvilíneas, onde as linhas de coordenadas constantes são as linhas da malha, conforme ilustrado na Figura 2.14. Estes métodos são mais onerosos que a maioria dos métodos algébricos e, portanto, não são apropriados para sistemas interativos, devido ao elevado tempo de resposta (Gonçalves, 2004).



**Figura 2.14:** Métodos Indiretos ou de Equações Diferenciais (Gonçalves, 2004).

Conforme Gonçalves (2004), num sistema gráfico interativo o processo de entrada de dados de geometria e topologia fica muito mais fácil e flexível, entretanto, um algoritmo deve fornecer respostas rápidas e eficientes para servir a este sistema. Portanto, visando atender estas exigências, os métodos de mapeamentos mais adequados são, segundo Fonseca (1989), os Mapeamentos Isoparamétrico e Transfinito.

A versão atual do gerador de malhas do sistema INSANE permite o emprego de Mapeamento Transfinito (Fonseca, 1989), Geração Nodal Aleatória (Ho-Le, 1988), Malha Assimétrica Concentrada e Malha com base em Grade (Ho-Le, 1988), sendo o primeiro uma técnica de mapeamento da fronteira, enquanto os demais são técnicas de decomposição de domínio. Estas técnicas foram implementadas por Ferreira e Pitangueira (2015), como parte de um projeto de iniciação científica.

## Capítulo 3

# METODOLOGIA DE INTEGRAÇÃO DO SISTEMA GRÁFICO

Conforme apresentado na seção 2.1, o sistema computacional INSANE é um ambiente para análise estrutural, desenvolvido em linguagem de programação JAVA, segundo o paradigma de Programação Orientada a Objetos, utilizando-se de diversas APIs e pacotes gráficos disponíveis para esta linguagem.

A entrada de dados neste sistema era realizada exclusivamente através de um arquivo texto escrito em Linguagem de Marcação eXtensível (XML), segundo regras previamente definidas, sendo o conhecimento da estrutura responsável pela leitura e interpretação deste arquivo uma condição necessária para o uso do sistema.

O aperfeiçoamento do pré-processador resultou em um ambiente gráfico de entrada de dados que, além de tornar a interação do sistema com o usuário mais amigável, propiciou um sistema gráfico integrado.

Para tanto, fez-se necessário estabelecer a comunicação entre os módulos desta aplicação, bem como com as outras duas grandes aplicações do sistema: o processador, a aplicação que representa o núcleo numérico do sistema e é o responsável pela obtenção dos resultados de diferentes modelos discretos de análise estrutural; e o pós-processador, uma aplicação gráfica interativa que disponibiliza ferramentas de visualização de resultados de diferentes modelos discretos.

A Estrutura de Dados para Subdivisão Planar, apresentada na seção 2.3, é responsável pelo armazenamento das entidades topológicas, o que permite a comunicação entre os módulos do pré-processador, cujos detalhes são apresentados no apêndice C. A cada mudança de módulo, quando da criação de sua interface, a estrutura de dados é clonada tal que preserve os dados de entrada relativos ao módulo a que pertence, permitindo retorno a estes módulos.

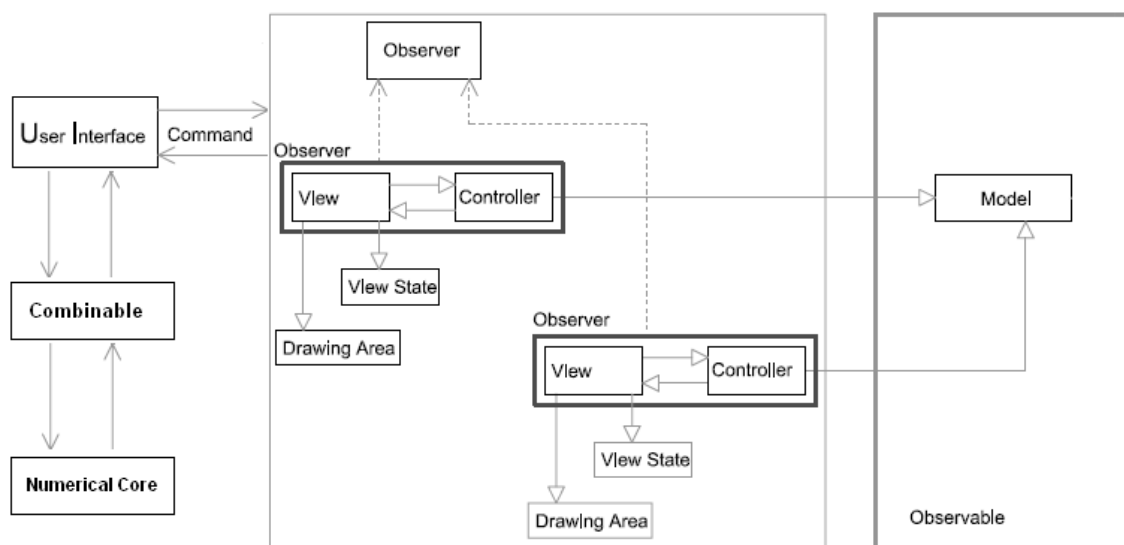
A comunicação do pré-processador com o processador é estabelecida a partir da adequação das entradas gráficas aos objetos de compreensão do núcleo numérico. Esta adequação é alcançada através da criação dos objetos *Model* e *Solution*, necessários para o processamento da análise. Estes objetos são criados a partir da topologia do modelo geométrico, armazenada na estrutura de dados, e das demais informações de entrada, tais como modelo de análise, modelo constitutivo, materiais, degenerações, carregamentos e parâmetros da solução, armazenadas em uma classe responsável pelo gerenciamento da análise.

Já a integração do pós-processador a este ambiente é possível a partir da escrita em XML dos resultados de todas as etapas do processamento, cuja leitura permite o preenchimento da estrutura de dados (do tipo “*Half-Edge*”) do Pós-Processador.

Muitos dos recursos necessários para implementação do pré-processador gráfico interativo no sistema INSANE já se encontravam disponíveis no código, sejam em implementações anteriores não usadas no então sistema atual, tais como aquelas apresentadas em Gonçalves (2004) e em Ferreira (2008), ou em implementações utilizadas em outras aplicações, tais como aquelas realizadas quando do desenvolvimento do pós-processador gráfico interativo, apresentadas em Penna (2007).

Conforme destacado na seção 2.4, geração de malhas baseada em mapeamento da fronteira e por meio de técnicas de decomposição de domínio são disponibilizadas no pré-processador, implementadas por Ferreira e Pitangueira (2015) aproveitando de implementações anteriores de Gonçalves (2004) e valendo de recursos implementados na aplicação de pós-processamento.

Consistentemente com a arquitetura básica do sistema INSANE discutida na seção 2.2, o pré-processador também utiliza uma combinação dos padrões de projeto *MVC*, *Command* e *Observer*. Esta combinação é apresentada na Figura 3.1, que ilustra vários pares vista-controlador observando o modelo. Conforme subseção 2.2.3, o controlador ativo é encarregado de disparar o processo de atualização da vista, substituindo o *Mecanismo de Propagação de Mudanças* do padrão *Observer*.



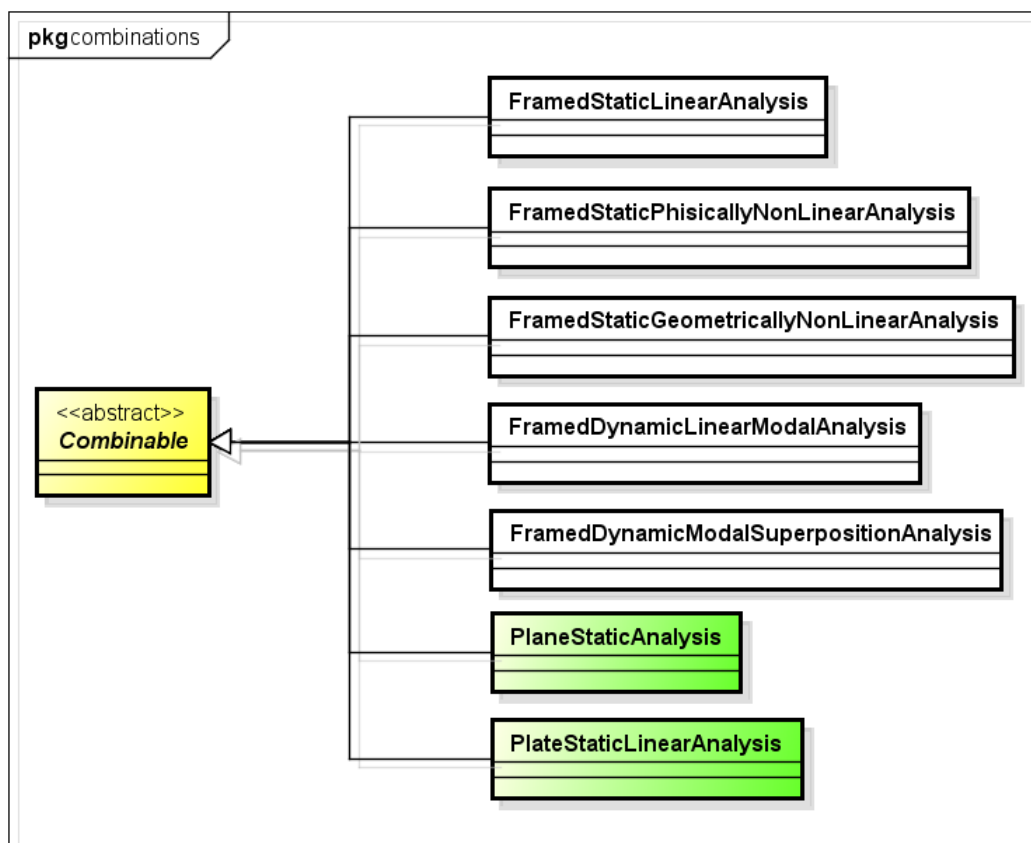
**Figura 3.1:** *MVC-Observer* (Adaptado de Penna (2007)).

Na Figura 3.1, destaca-se o núcleo numérico que abastece a interface com o usuário em função do tipo de análise, através de uma classe responsável por intermediar a comunicação da aplicação gráfica de entrada de dados com o núcleo numérico, denotada de forma abstrata por *Combinable*.

No ambiente gráfico, ao se definir o tipo de análise, define-se a classe especializada da classe *Combinable*, responsável pelo gerenciamento da análise. Esta classe combina os parâmetros aplicáveis à análise que se destina, filtrados do núcleo numérico, disponibilizando na interface com o usuário aqueles que requerem escolha e/ou determinação de seus campos. Esta combinação de parâmetros deu origem ao nome *Combinable*. Dentre estes parâmetros, destacam-se modelos de análise, modelos constitutivos, tipos de material, tipos de degeneração e processos de solução.



Logo, para cada tipo de análise disponibilizada no sistema gráfico integrado, tem-se associada uma classe especializada da classe *Combinable*. O diagrama UML (*Unified Modelling Language*) apresentado na Figura 3.2 ilustra a estrutura hierárquica da classe *Combinable*. As classes que foram modificadas estão identificadas em amarelo, enquanto as classes em verde representam aquelas que foram criadas.



**Figura 3.2:** Estrutura hierárquica da classe *Combinable*.

No estágio atual do sistema INSANE, permitem-se análises unidimensionais e bidimensionais através do sistema gráfico integrado, cujo ambiente é específico em função da classificação espacial da análise, preservando, porém, a generalidade do programa. As classes especializadas da classe *Combinable* não criadas no desenvolvimento deste trabalho referem-se às opções de análise unidimensional e, portanto, são disponibilizadas no ambiente gráfico desta análise, o qual foi recuperado e integrado como adendo à este trabalho.

A classe *PlaneStaticAnalysis* foi desenvolvida para gerenciamento de análises bidimensionais estáticas lineares ou fisicamente não lineares com carregamento no plano pelo MEF ou pelo MEEG, ao passo que a classe *PlateStaticLinearAnalysis* para gerenciamento de análises bidimensionais estáticas lineares com carregamento normal ao plano (placas) pelo MEF.

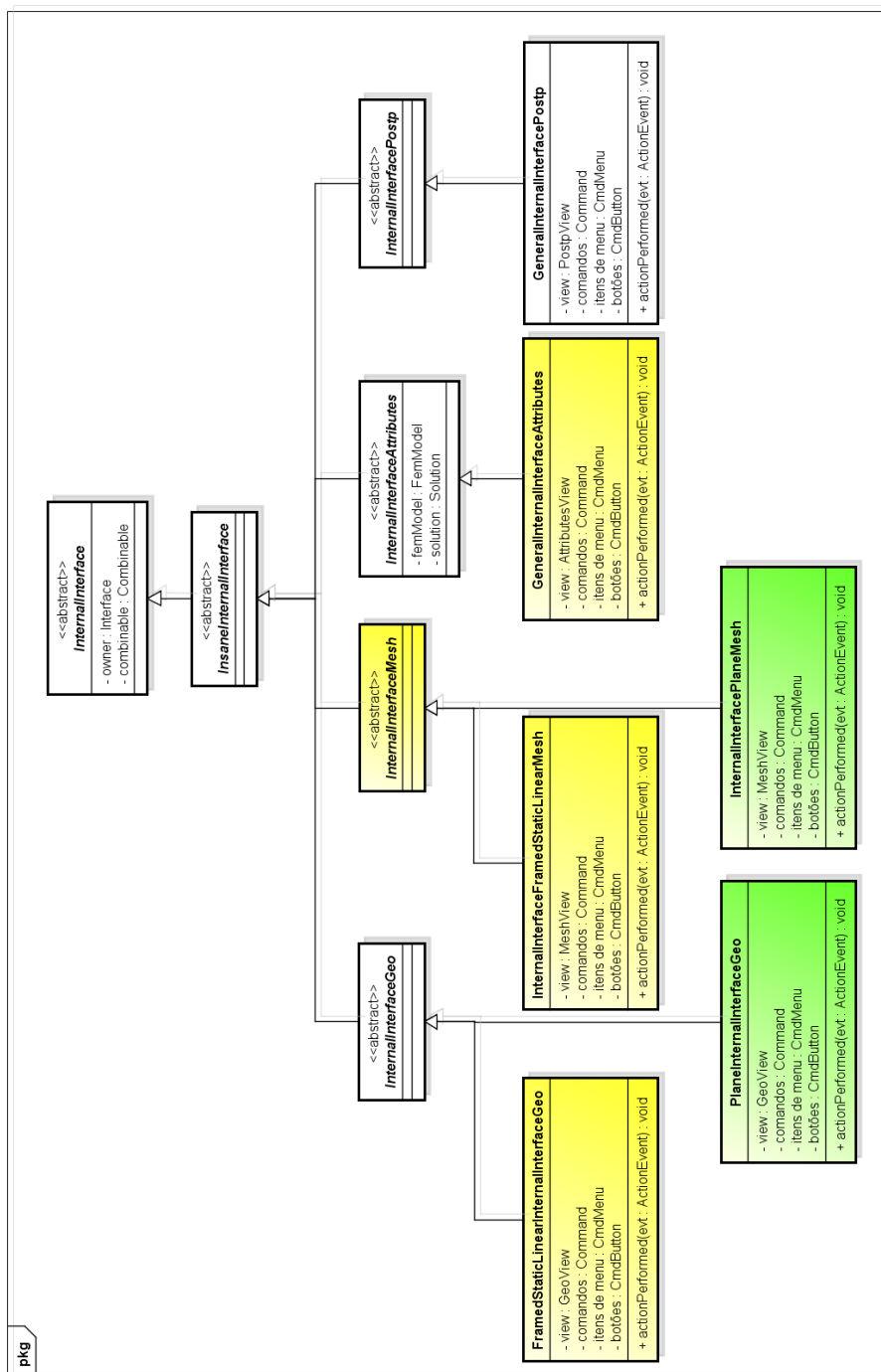
Da perspectiva da implementação, conforme Figura 3.3, o objeto *Combinable* é uma variável da classe abstrata *InternalInterface*, a partir da qual tem-se especializadas as diversas interfaces de interação com o usuário do sistema INSANE.

Conforme detalhado no apêndice C, o ambiente gráfico de entrada de dados é dividido em módulos. A entrada no sistema gráfico se dá definindo o tipo de análise segundo classificação espacial, ao passo que a especialização da análise, e consequente construção do objeto *Combinable*, é definida na mudança para o módulo a partir do qual esta informação é necessária para especialização das interfaces.

Esta especialização consiste na determinação dos recursos disponibilizados para aquela análise, realizada segundo o tipo do objeto *Combinable*, e na definição das rotinas de execução acionadas pelos comandos. Conforme Figura 3.3, as interfaces possuem como variáveis diversos objetos *Command*, cujo argumento para construção é fundamentalmente a própria interface que os contém. Desta forma, os comandos, que são associados aos elementos de interface (botões e menus), têm acesso ao objeto *Combinable* e, conseqüentemente, aos parâmetros da análise, permitindo que as rotinas de execução disparadas através destes comandos sejam específicas e/ou levem em consideração as características da análise.

Desta forma, não foi necessária a implementação de interfaces específicas para cada tipo de análise, resultando na generalidade do ambiente gráfico de entrada de dados, sendo as especificidades de cada análise tratadas através do *Combinable*.

O diagrama UML apresentado na Figura 3.3 ilustra as principais interfaces do sistema. Novamente, as classes que foram modificadas estão identificadas em amarelo, enquanto as classes em verde representam aquelas que foram criadas.



**Figura 3.3:** Estrutura hierárquica das principais interfaces do sistema INSANE.

Através da Figura 3.3, é possível se ter uma ideia do fluxo de utilização do sistema gráfico integrado. Este fluxo é também observado na Figura 3.4, onde se ilustra a estrutura gráfica do ambiente de análise bidimensional do INSANE, destacando-se os principais agentes interativos empregados na integração deste sistema.

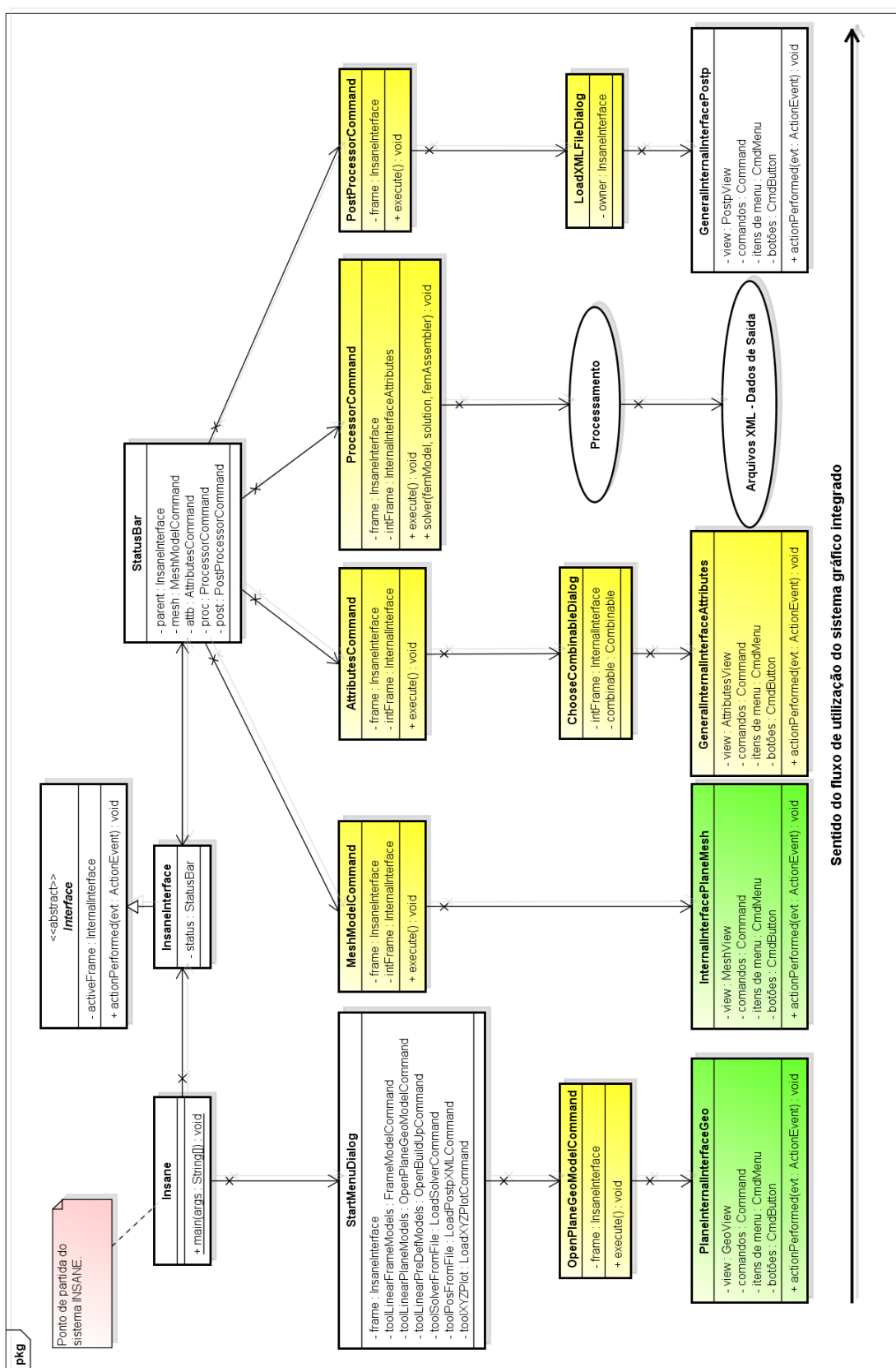


Figura 3.4: Estrutura gráfica do ambiente de análise bidimensional do INSANE.

Conforme Figura 3.4, o ponto de partida do sistema INSANE, estabelecido pelo método estático *main(String[] args)*, encontra-se em uma classe que leva o nome deste sistema. A partir da classe *Insane*, tem-se criada uma instância da classe *InsaneInterface*, especializada da classe abstrata *Interface*, que consiste na janela principal deste sistema. À esta interface, é atribuída a responsabilidade pelo gerenciamento do fluxo de utilização do ambiente gráfico integrado.

O objeto *InsaneInterface* tem como uma de suas variáveis uma instância da classe *StatusBar*, conforme Figura 3.4, responsável por gerenciar a liberação dos botões de fluxo de utilização no ambiente gráfico em função do módulo corrente.

Após a exibição da interface *InsaneInterface*, tem-se a exibição do diálogo inicial do sistema, instância da classe *StartMenuDialog*, também criado através da classe *Insane*. Este diálogo é composto por botões que definem as aplicações do sistema. No estágio atual, tem-se as aplicações para análise unidimensional, bidimensional e de modelos pré-definidos via sistema gráfico integrado, a aplicação para análise unidimensional, bidimensional e tridimensional via arquivo XML e a aplicação para visualização gráfica de resultados, arquivados ao final de um processamento.

A aplicação para análise bidimensional via ambiente gráfico encapsula o comando *OpenPlaneGeoModelCommand*, cujo método *execute()* é responsável pela construção e abertura da interface *PlaneInternalInterfaceGeo* (Figura 3.4). Na construção desta interface, tem-se a criação de sua vista (*GeoView*) que, por sua vez, instancia o correspondente controlador (*GeoPrepController*), o qual inicializa a estrutura de dados (*HalfEdgeDataStructure*) responsável por armazenar as entidades topológicas do modelo geométrico a ser construído (seção 2.3).

O objeto *StatusBar* é acionado quando da construção desta interface, disponibilizando o botão que encapsula o comando *MeshModelCommand*, a partir do qual tem-se a mudança para o próximo módulo do ambiente de entrada de dados.

Ao se acionar este comando, tem-se a construção e abertura da interface *InternalInterfacePlaneMesh*.

A esta interface, associa-se a estrutura de dados obtida pela clonagem da estrutura de dados do módulo anterior, armazenando-a no controlador (*MeshController*) pertencente à vista (*MeshView*) desta interface.

O objeto *StatusBar* é novamente acionado na construção desta interface, disponibilizando o botão que encapsula o comando *AttributesCommand*, a partir do qual tem-se a mudança para o último módulo do ambiente de entrada de dados.

Esta mudança é intermediada pelo diálogo *ChooseCombinableDialog*, criado e aberto pela classe *AttributesCommand*, no qual tem-se a especialização da análise.

Ao se definir a especialização da análise, tem-se a criação do respectivo *Combinable*, armazenado na classe abstrata *InternalInterface* (Figura 3.3).

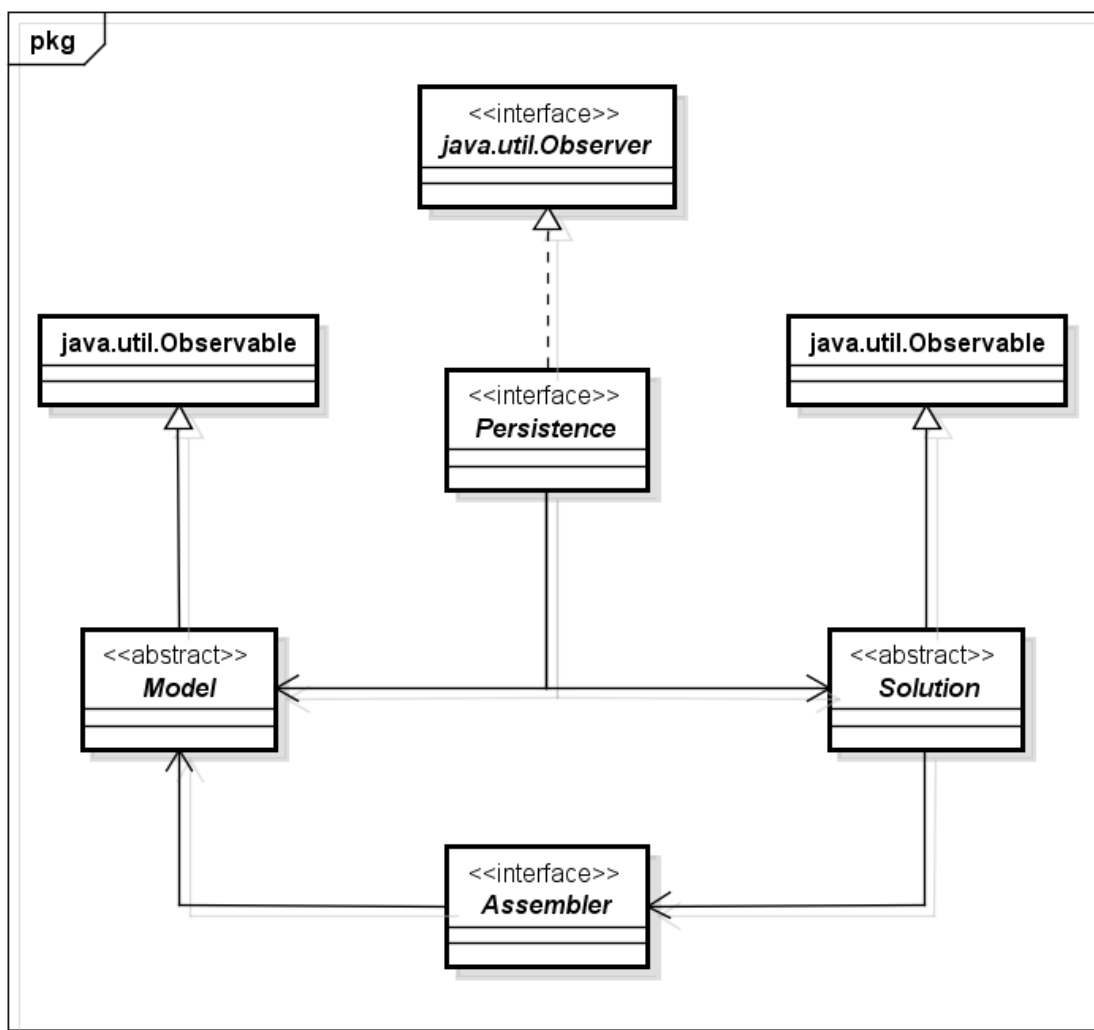
À classe *ChooseCombinableDialog*, é também atribuída a construção e abertura da interface *GeneralInternalInterfaceAttributes*.

A esta interface, associa-se a estrutura de dados obtida pela clonagem da estrutura de dados do módulo anterior, armazenando-a no controlador (*AttributesController*) pertencente à vista (*AttributesView*) desta interface.

No módulo gerenciado por esta interface, tem-se disponibilizado pelo objeto *StatusBar* o botão que encapsula o comando *ProcessorCommand*, responsável por enviar os objetos *Model* e *Solution* ao núcleo numérico para processamento da análise. Ao final do processamento, o objeto *StatusBar* disponibiliza neste mesmo módulo o botão que encapsula o comando *PostProcessorCommand*, através do qual tem-se a criação do diálogo implícito *LoadXMLFileDialog*, responsável pelo carregamento dos arquivos XML dos resultados de todas as etapas do processamento, gerados pelo núcleo numérico, para abastecimento do pós-processador.

Na Figura 3.4, destaca-se o método *solver(femModel, solution, femAssembler)* da classe *ProcessorCommand*, responsável por disparar o processamento da análise. Este método tem como argumentos uma instância de uma classe que estende a classe abstrata *Model*, uma instância de uma classe que estende a classe abstrata *Solution* e uma instância de uma classe que implementa a interface *Assembler*.

As classes abstratas *Model* e *Solution* e a interface *Assembler*, juntamente com a interface *Persistence*, compõem o núcleo numérico do sistema INSANE. Através do diagrama de classes em UML ilustrado na Figura 3.5 é possível identificar estas classes, bem como a comunicação entre elas.



**Figura 3.5:** Organização do núcleo numérico do INSANE.

Conforme Penna (2011), a classe abstrata *Model*, a interface *Assembler* e a classe abstrata *Solution*, respectivamente, compõem as abstrações do modelo discreto adotado, a generalização do processo de montagem do sistema matricial de equações e as técnicas de solução necessárias à obtenção dos resultados da análise.

A interface *Persistence*, por sua vez, auxilia o núcleo numérico, responsabilizando-se pelo tratamento dos dados não voláteis do programa, ou seja, as classes que formam a hierarquia de *Persistence* são responsáveis pelo gerenciamento dos dados da análise e pela tarefa destacada por vezes neste capítulo da geração de dados gravados em disco, sob a forma de arquivos XML, para a posterior leitura dos resultados pelo pós-processador. No caso de análise cuja entrada é via arquivo XML, a interface *Persistence* é também responsável por coletar os dados de entrada oriundos destes arquivos para construção dos objetos *Model* e *Solution*, dentre outros necessários para o processamento da análise.

A tarefa de construção dos objetos *Model* e *Solution*, no caso de análise via sistema gráfico integrado, é atribuída à classe *Combinable*, por ser esta classe responsável pelo acesso do ambiente gráfico ao núcleo numérico, conforme discutido.

Através do método *execute()* do comando *ProcessorCommand*, tem-se disparado o método *execute()* do comando *BuildFemModelCommand*, responsável por solicitar a construção do objeto *Model* à classe *Combinable*, armazenando-o na interface *InternalInterfaceAttributes*, conforme destacado na Figura 3.3.

Na sequência, o comando *ProcessorCommand* cria e abre o diálogo *ProcessorDialog*, através do qual tem-se a entrada dos parâmetros da solução. Com o aceite, tem-se a definição do objeto *Assembler* em função do objeto *Model* e a solicitação da construção do objeto *Solution* à classe *Combinable*, armazenando-o, também, na interface *InternalInterfaceAttributes*, conforme destacado na Figura 3.3, ao passo que o objeto *Assembler* é armazenado no objeto *Solution*.

Na continuidade do método *execute()* do comando *ProcessorCommand*, buscam-se na interface *InternalInterfaceAttributes* os objetos *Model* e *Solution*, bem como o objeto *Assembler* no objeto *Solution*, finalizando este método e a etapa de pré-processamento ao disparar o método privado *solver(...)*.

Para melhor compreensão das tarefas atribuídas à classe *Combinable*, apresentam-se no apêndice B suas principais variáveis e seus principais métodos.



## Capítulo 4

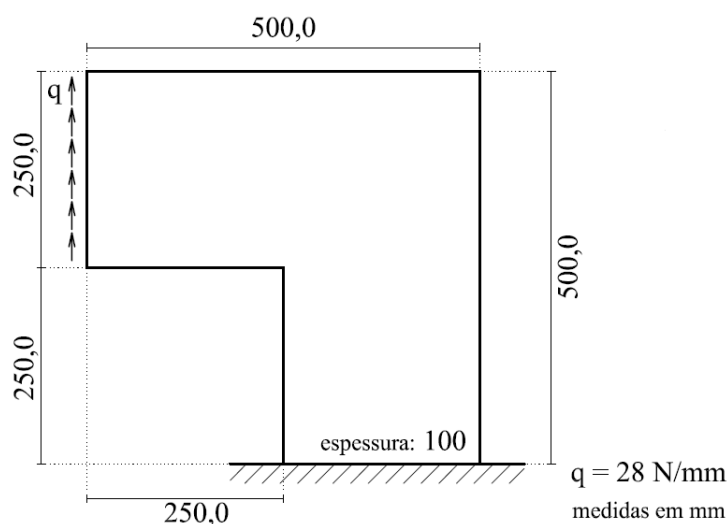
# EXEMPLOS DE MODELAGEM NO SISTEMA GRÁFICO INTEGRADO

Neste capítulo, apresentam-se 6 exemplos de modelagem no sistema gráfico integrado. Cada exemplo foi simulado visando destacar uma aplicação deste sistema para a opção de análise bidimensional estática. Os exemplos, bem como os objetivos de suas simulações, são listados a seguir:

- Seção 4.1: Painel em “L” - Objetivou-se destacar a opção de análise linear com carregamento no plano pelo MEF;
- Seção 4.2: Chapa com trincas em modo I de abertura - Objetivou-se destacar a opção de análise linear com carregamento no plano pelo MEF;
- Seção 4.3: Barragem submetida à carga hidrostática - Objetivou-se destacar a opção de análise fisicamente não linear com carregamento no plano pelo MEF;
- Seção 4.4: Viga pré-tensionada sob flexão - Objetivou-se destacar a opção de análise fisicamente não linear com carregamento no plano pelo MEF;
- Seção 4.5: Viga com pré-trinca sob flexão - Objetivou-se demonstrar a capacidade do recurso de modelagem de descontinuidades preexistentes na estrutura;
- Seção 4.6: Placa anular com carga na borda interna - Objetivou-se destacar a opção de análise linear com carregamento normal ao plano (placas) pelo MEF.

## 4.1 Painel em “L”

O painel em “L”, cujas geometria, vinculação e carga são ilustradas na Figura 4.1, foi simulado no sistema gráfico integrado objetivando-se destacar o emprego deste sistema para a opção de análise bidimensional estática linear com carregamento no plano pelo MEF.



**Figura 4.1:** Painel em “L”: Configuração (Penna, 2011).

Para esta modelagem, adotaram-se os seguintes parâmetros:

- Opção de análise: Carregamento no plano
- Técnica de geração de malhas: Geração Nodal Aleatória

Dimensão média dos elementos: 50 *mm*

- Refinamento local da malha:

Região: Faixa horizontal a partir da junção angular

Largura desta região: 150 *mm*

Relação de definição da dimensão média dos elementos nesta região: 0,2

Nível de suavização: 1,0

- Modelo de análise: Estado Plano de Tensão
- Elementos finitos: T6
- Modelo constitutivo: Linear elástico
- Materiais:

Material 1:

Tipo: Linear elástico isotrópico

Parâmetros:

Módulo de elasticidade:  $E = 25850,0 \text{ N/mm}^2$

Coefficiente de Poisson:  $\nu = 0,18$

- Degenerações:

Seção 1:

Tipo: Degeneração prescrita

Propriedades:

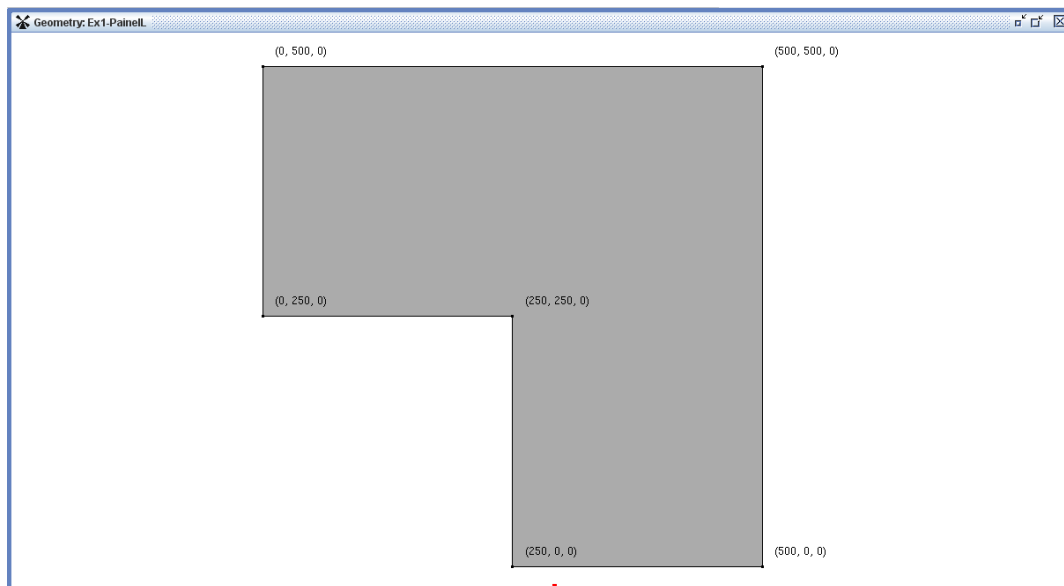
Espessura:  $100 \text{ mm}$

Material associado: Material 1

- Carregamentos:

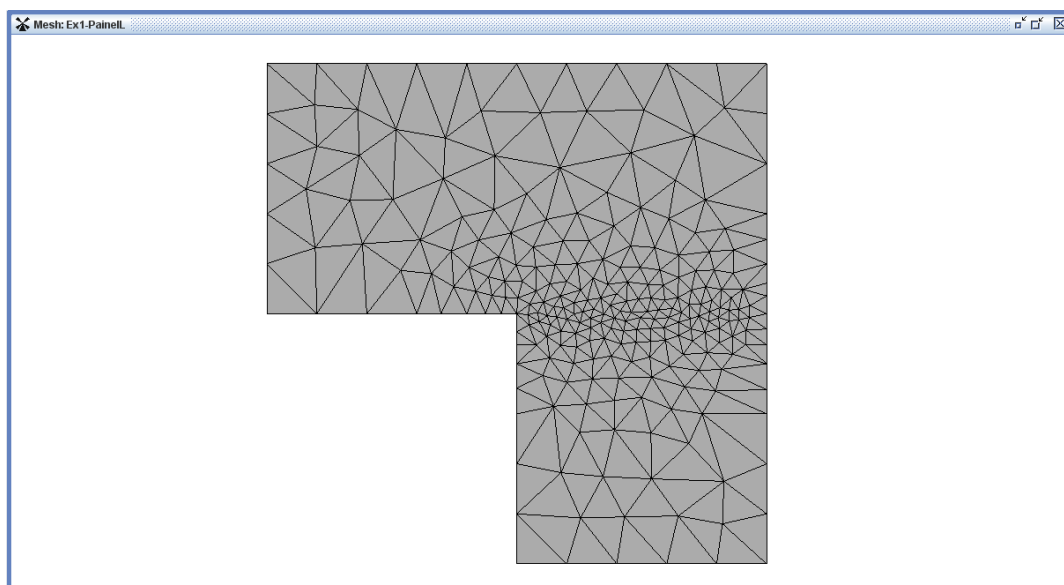
Não incremental:  $q = 28 \text{ N/mm}$

Conforme discutido no apêndice C, a modelagem via sistema gráfico integrado se inicia desenhando-se a estrutura através do Módulo *Geometria*. Para este exemplo, bastou-se desenhar a face definida pelo contorno do painel em “L”, conforme apresentado na Figura 4.2. Tendo em vista a opção pela Geração Nodal Aleatória como a técnica de geração de malhas, não foi necessário decompor a geometria da estrutura em regiões triangulares e/ou quadrilaterais.



**Figura 4.2:** Painel em “L”: Geometria.

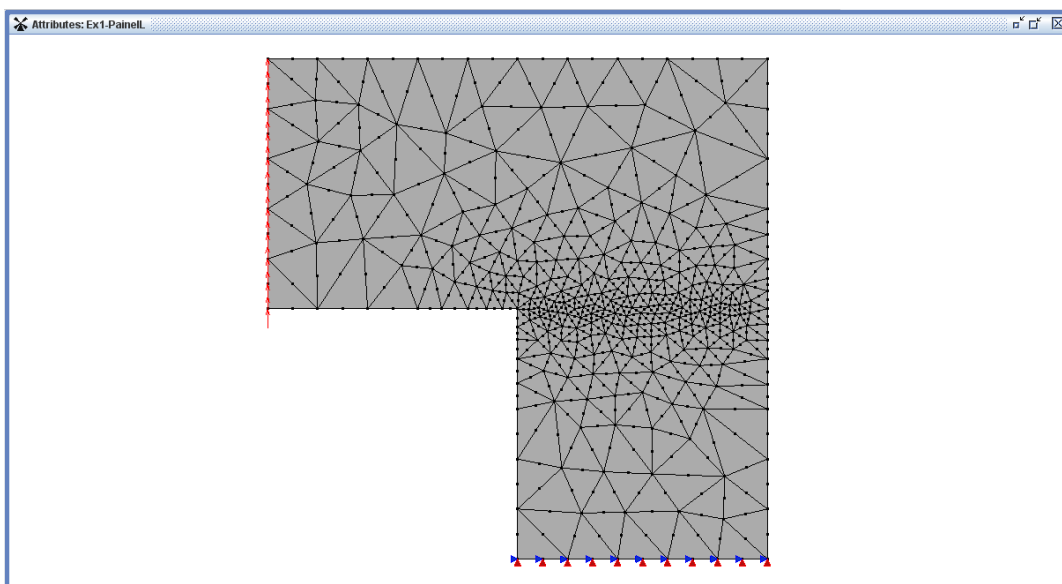
Na sequência, através do Módulo *Malha*, discretizou-se a geometria da estrutura por meio de técnica de decomposição de domínio, conforme apresentado na Figura 4.3. Visando aumentar a densidade de malha na faixa horizontal a partir da junção angular do painel em “L”, utilizou-se o recurso de refinamento local da malha.



**Figura 4.3:** Painel em “L”: Malha.

Após a discretização da geometria, bem como a definição do modelo de análise, seguiu-se para o Módulo *Atributos*. Na passagem para este módulo, definiu-se também o tipo de elemento finito a se empregar na análise.

No Módulo *Atributos*, definiram-se as condições de contorno e o modelo constitutivo e criaram-se o material e a degeneração. Este material foi associado a esta degeneração, ao passo que esta degeneração foi atribuída a todos os elementos do modelo. Na Figura 4.4, apresenta-se a geometria da estrutura discretizada com as condições de contorno, destacando-se também o tipo de elemento finito.

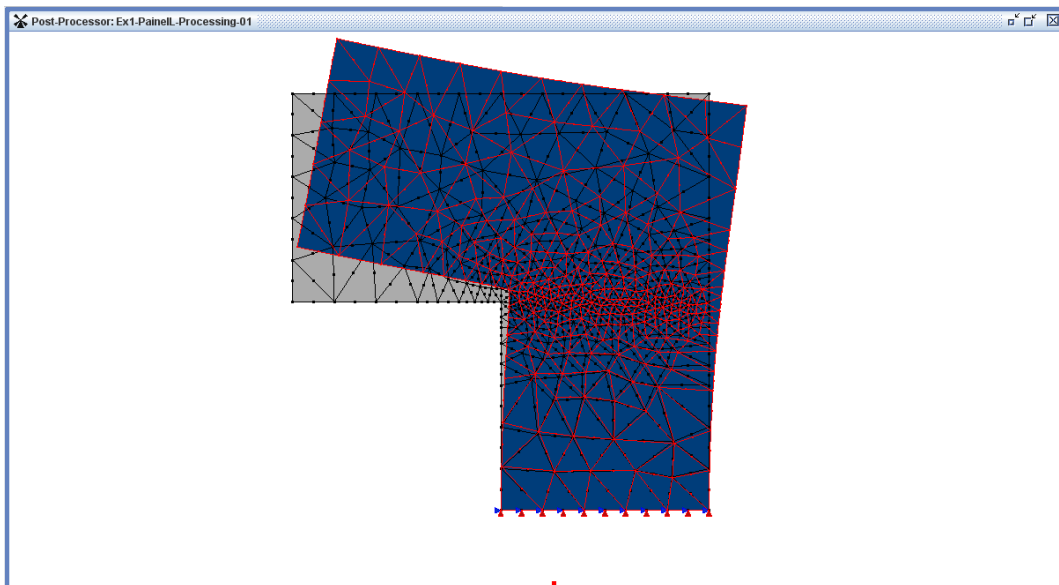


**Figura 4.4:** Painel em “L”: Condições de contorno.

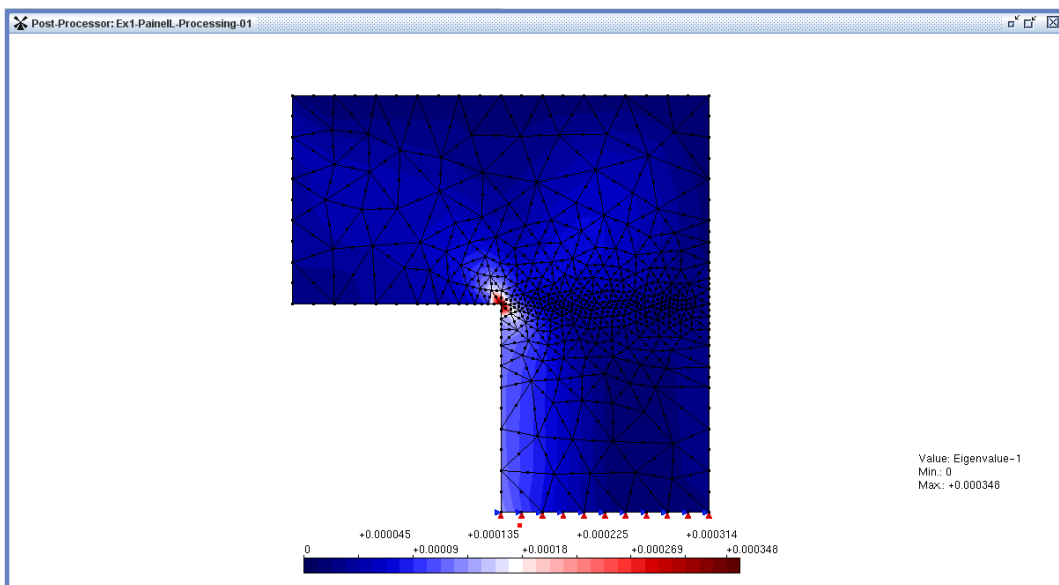
Após a definição dos atributos do modelo, processou-se a análise. Destaca-se que, por se tratar de uma análise linear, não foi necessária a definição de parâmetros da técnica de solução.

Ao final do processamento, seguiu-se para o pós-processador. O estudo das respostas de uma estrutura quando solicitada por agentes externos é realizado através desta aplicação que, por meio dos seus diversos recursos para visualização gráfica dos resultados, permite avaliar a variação de grandezas de estado ao longo do domínio da estrutura.

Para o painel em “L”, por exemplo, avaliaram-se sua deformada, apresentada na Figura 4.5 com fator de escala igual a 500, e a variação da deformação  $\varepsilon_1$  ao longo do seu domínio, apresentada na Figura 4.6.



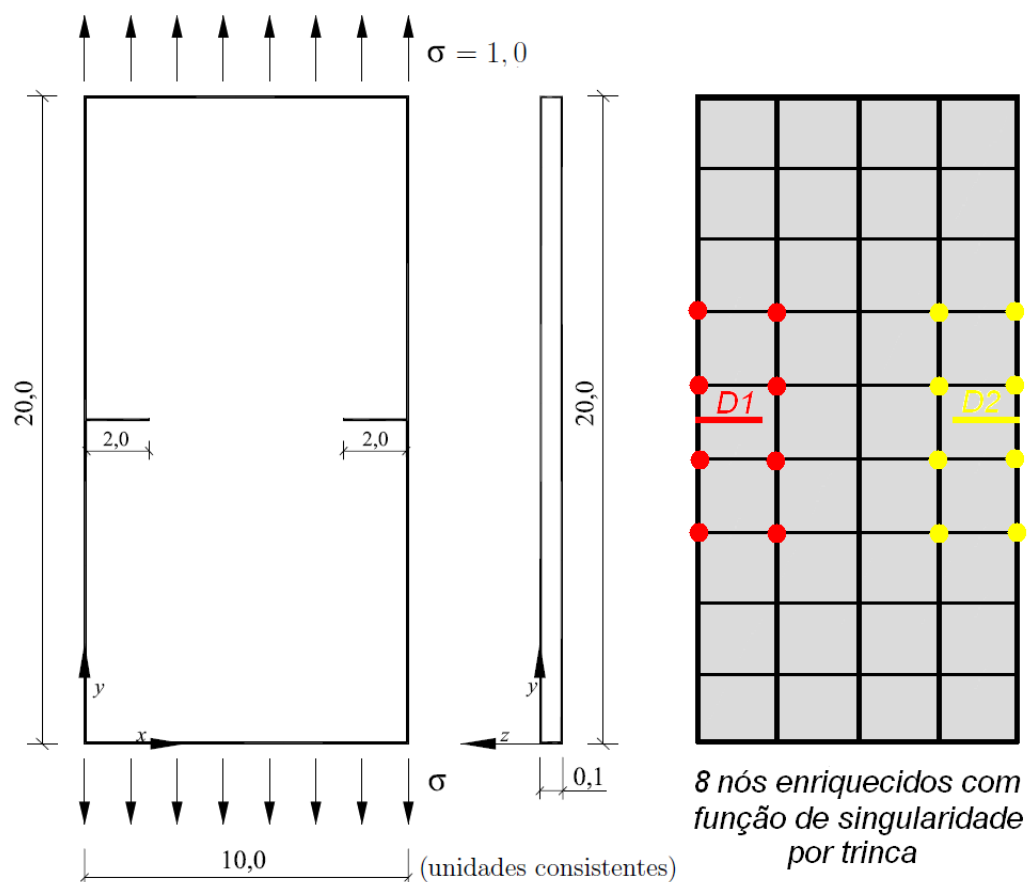
**Figura 4.5:** Painel em “L”: Deformada.



**Figura 4.6:** Painel em “L”: Deformação  $\varepsilon_1$ .

## 4.2 Chapa com Trincas em Modo I de Abertura

A chapa com trincas em modo I de abertura, cujas geometria e cargas são ilustradas na Figura 4.7, foi simulada no sistema gráfico integrado objetivando-se destacar o emprego deste sistema para a opção de análise bidimensional estática linear com carregamento no plano pelo MEFG.



**Figura 4.7:** Chapa com trincas: Configuração.

Na Figura 4.7, destacam-se os nós enriquecidos com função de singularidade. Observa-se que cada conjunto de nós foi enriquecido com a função correspondente a uma trinca. Esta função de enriquecimento, criada e atribuída aos nós de interesse pelo analista, potencializa a simulação do fenômeno de singularidade no campo de tensões observado em regiões próximas à ponta de trincas. Esta atribuição de funções de enriquecimento aos nós condiciona a análise ao MEFG.

Nesta simulação, as trincas foram modeladas através do recurso que permite considerá-las sem a necessidade de introduzi-las na geometria do modelo. A partir deste recurso, as descontinuidades preexistentes são simuladas por meio de funções degrau, que também consistem em funções de enriquecimento nodal, porém, criadas e atribuídas aos nós necessários pelo programa, cabendo ao analista apenas a definição dos pontos inicial e final destas descontinuidades.

Por serem simuladas por meio de funções de enriquecimento nodal, a existência destas descontinuidades no modelo, por si só, já condiciona a análise ao MEF-G.

O recurso de modelagem de descontinuidades preexistentes é melhor avaliado através do exemplo da seção 4.5, que teve por objetivo analisar a resposta obtida empregando-se este recurso e demonstrar a sua capacidade.

Para esta modelagem, adotaram-se os seguintes parâmetros:

- Opção de análise: Carregamento no plano
- Técnica de geração de malhas: Mapeamento Transfinito
- Modelo de análise: Estado Plano de Tensão
- Elementos finitos: Q4
- Modelo constitutivo: Linear elástico
- Materiais:

Material 1:

Tipo: Linear elástico isotrópico

Parâmetros:

Módulo de elasticidade:  $E = 1,0$

Coefficiente de Poisson:  $\nu = 0,3$



- Degenerações:

Seção 1:

Tipo: Degeneração prescrita

Propriedades:

Espessura: 0,1

Material associado: Material 1

- Funções de enriquecimento nodal:

Enriquecimento 1 (correspondente à trinca D1):

Tipo: Singularidade

Parâmetros:

Ponta da trinca - coordenada  $x$ : 2,0

Ponta da trinca - coordenada  $y$ : 10,0

Parâmetro  $\lambda$  da função de singularidade: 0,5

Parâmetro  $Q$  da função de singularidade: 0,333333333

Ângulo entre as direções  $x$  e de crescimento da trinca:  $\theta = 0,0$

Enriquecimento 2 (correspondente à trinca D2):

Tipo: Singularidade

Parâmetros:

Ponta da trinca - coordenada  $x$ : 8,0

Ponta da trinca - coordenada  $y$ : 10,0

Parâmetro  $\lambda$  da função de singularidade: 0,5

Parâmetro  $Q$  da função de singularidade: 0,333333333

Ângulo entre as direções  $x$  e de crescimento da trinca:  $\theta = \pi$

- Descontinuidades (entanches):

Descontinuidade 1 (trinca D1):

Tipo: Simulada pelo MEFG

Parâmetros:

Ponto inicial ( $x y$ ): (0,0 10,0)

Ponto final ( $x y$ ): (2,0 10,0)

Descontinuidade 2 (trinca D2):

Tipo: Simulada pelo MEFG

Parâmetros:

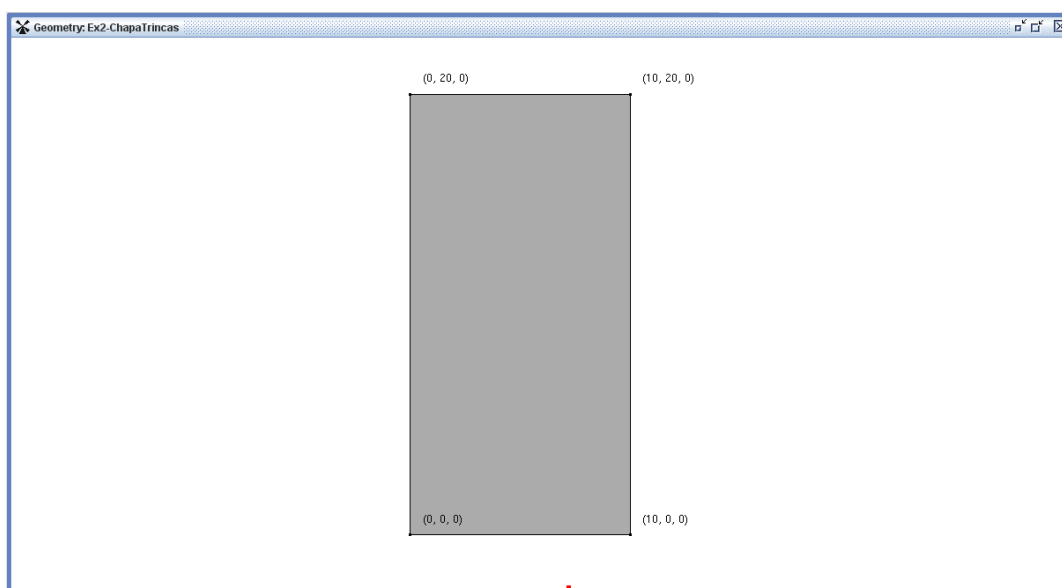
Ponto inicial ( $x y$ ): (8,0 10,0)

Ponto final ( $x y$ ): (10,0 10,0)

- Carregamentos:

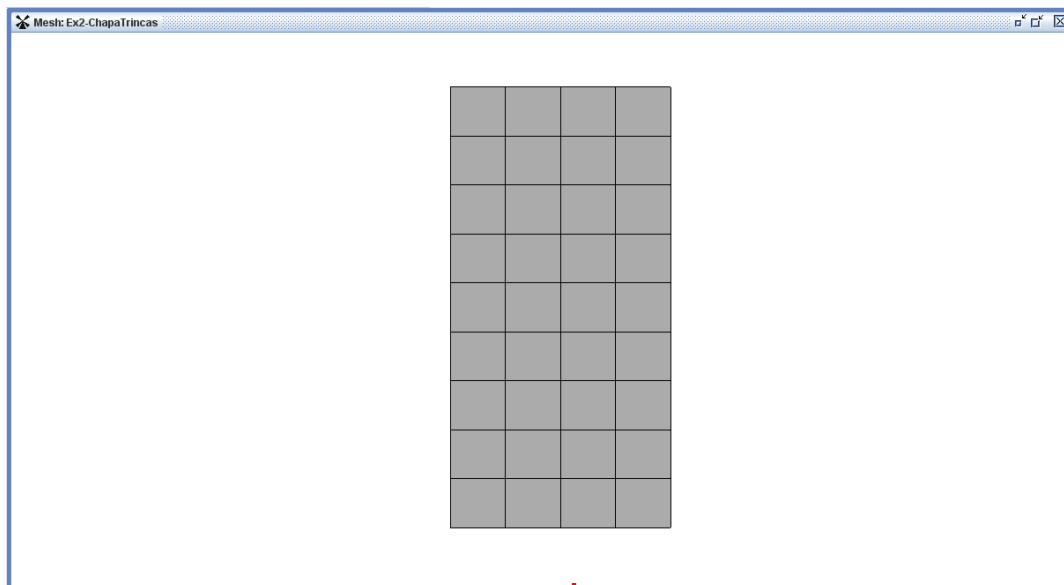
Não incremental:  $\sigma = 1,0$  aplicada como força uniforme de  $f_y = 0,1$

No Módulo *Geometria*, desenhou-se a geometria da estrutura, conforme apresentado na Figura 4.8.



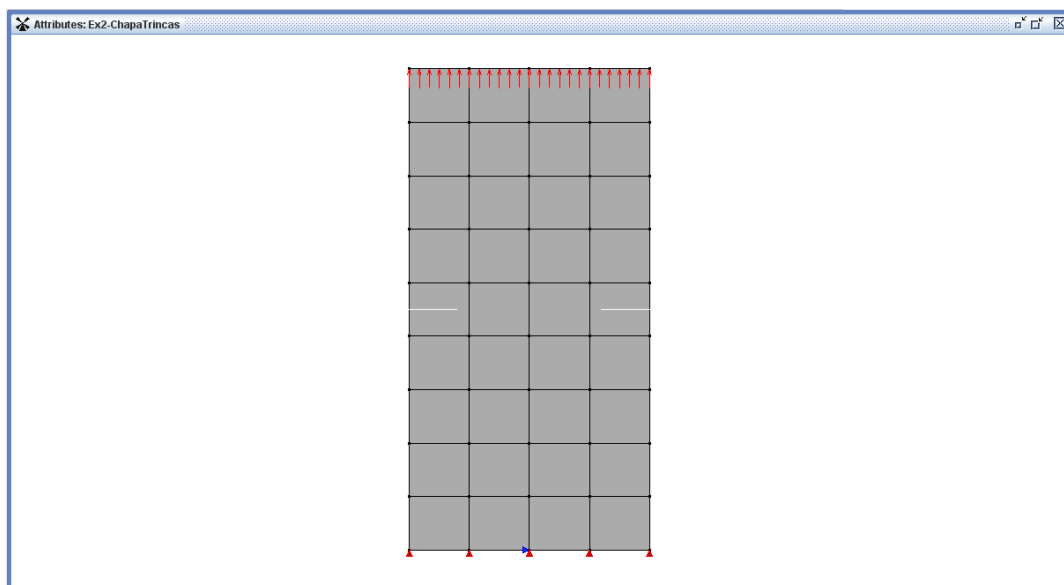
**Figura 4.8:** Chapa com trincas: Geometria.

No Módulo *Malha*, discretizou-se a geometria da estrutura, conforme apresentado na Figura 4.9, e definiram-se o modelo de análise e o tipo de elemento finito.



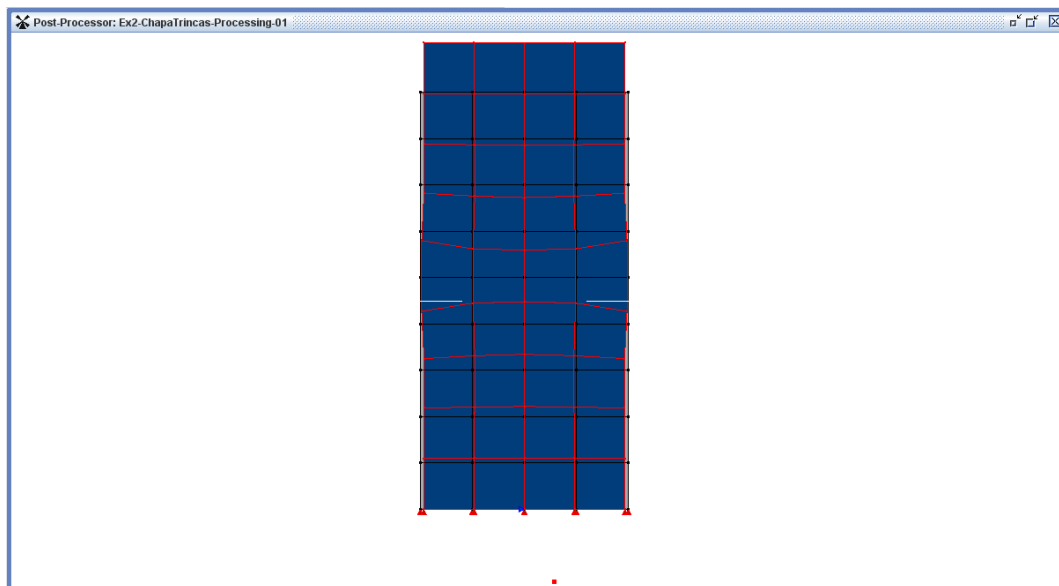
**Figura 4.9:** Chapa com trincas: Malha.

No Módulo *Atributos*, definiram-se as condições de contorno e o modelo constitutivo, criaram-se e atribuíram-se o material, a degeneração e as funções de enriquecimento e modelaram-se as descontinuidades, observadas na Figura 4.10.

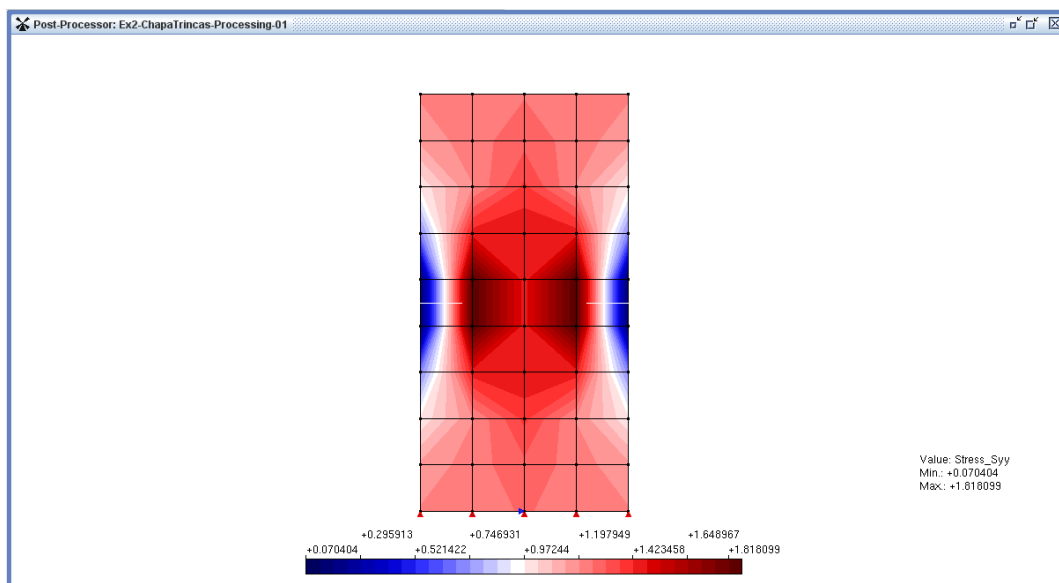


**Figura 4.10:** Chapa com trincas: Condições de contorno.

No pós-processador, avaliaram-se os saltos no campo de deslocamentos provocados pela presença das trincas, observados através da deformada apresentada na Figura 4.11 com fator de escala igual a 0,1, bem como a magnitude do campo de tensões nas regiões próximas às pontas das trincas, observada através da variação da tensão  $\sigma_{yy}$  ao longo do domínio apresentada na Figura 4.12.



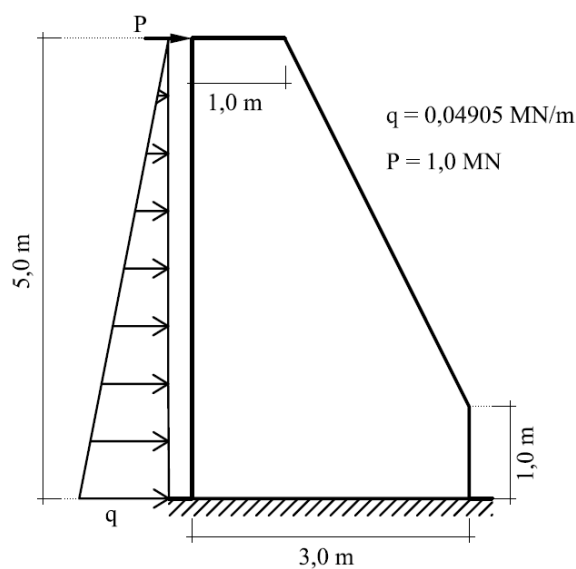
**Figura 4.11:** Chapa com trincas: Deformada.



**Figura 4.12:** Chapa com trincas: Tensão  $\sigma_{yy}$ .

### 4.3 Barragem Submetida à Carga Hidrostática

A barragem submetida à carga hidrostática, cujas geometria, vinculação e cargas são ilustradas na Figura 4.13, foi reproduzida de Penna (2011) no sistema gráfico integrado objetivando-se destacar o emprego deste sistema para a opção de análise bidimensional estática fisicamente não linear com carregamento no plano pelo MEF.



**Figura 4.13:** Barragem: Configuração (Penna, 2011).

Para esta modelagem, adotaram-se os seguintes parâmetros:

- Opção de análise: Carregamento no plano
- Técnica de geração de malhas: Malha com base em Grade

Dimensão média dos elementos: 0,5 m

Algoritmo gerador da malha: quadrilateral

- Modelo de análise: Estado Plano de Deformação
- Elementos finitos: T3
- Modelo constitutivo: Dano volumétrico por Penna (2011)

- Materiais:

Material 1:

Tipo: Inerente ao modelo constitutivo

Parâmetros:

Módulo de elasticidade:  $E = 30000,0 \text{ MPa}$

Coefficiente de Poisson:  $\nu = 0,2$

Lei de evolução de dano para tração:

Tipo: Polinomial

Parâmetros:

Deformação equivalente:  $\kappa_0 = 0,0002825$

Limite de resistência equivalente:  $f_e = 1,85 \text{ MPa}$

Módulo de elasticidade equivalente:  $\tilde{E} = 16666,666 \text{ MPa}$

Lei de evolução de dano para compressão:

Tipo: Polinomial

Parâmetros:

Deformação equivalente:  $\kappa_0 = 0,005$

Limite de resistência equivalente:  $f_e = 11,1 \text{ MPa}$

Módulo de elasticidade equivalente:  $\tilde{E} = 16666,666 \text{ MPa}$

- Degenerações:

Seção 1:

Tipo: Degeneração prescrita

Propriedades:

Espessura:  $1,0 \text{ m}$  (segmento unitário)

Material associado: Material 1

- Carregamentos:

Não incremental:  $q = 0,04905 \text{ MN}/m$  (carga triangular)

Incremental:  $P = 1,0 \text{ MN}$

- Parâmetros da técnica de solução:

Processo incremental-iterativo: Newton-Raphson Padrão

Método de controle: Deslocamento generalizado

Fator de carga inicial: 0,1

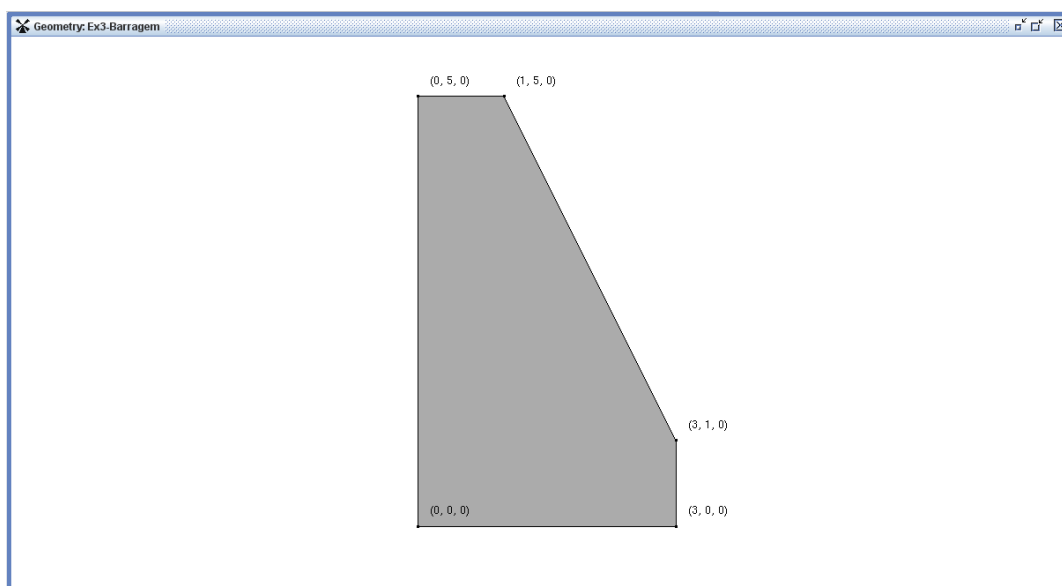
Número máximo de passos: 500

Número máximo de iterações: 100

Tolerância:  $1,0 \times 10^{-4}$

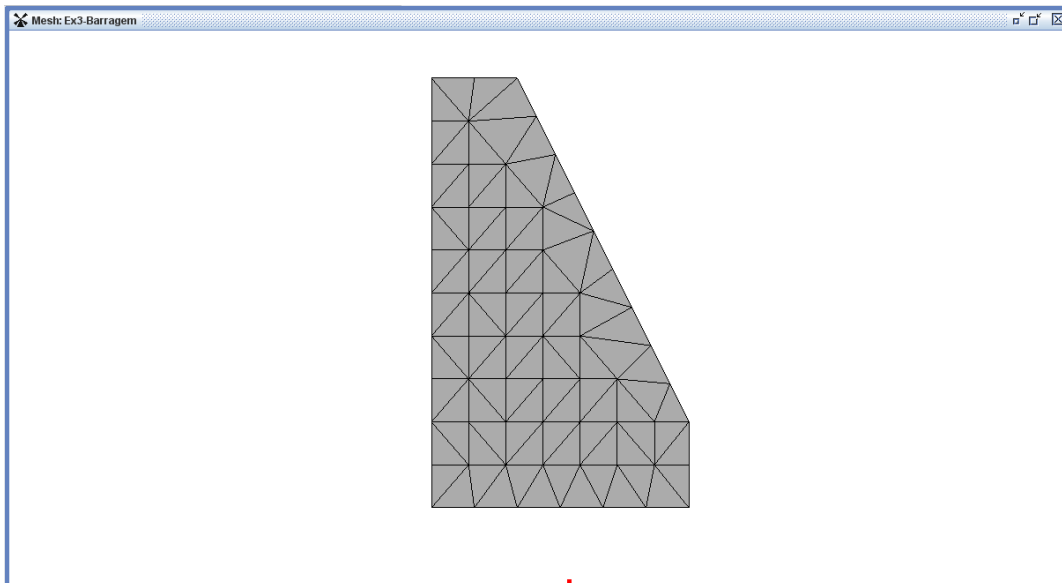
Tipo de convergência: Magnitude do vetor de deslocamentos incrementais

No Módulo *Geometria*, desenhou-se a geometria da estrutura, conforme apresentado na Figura 4.14.



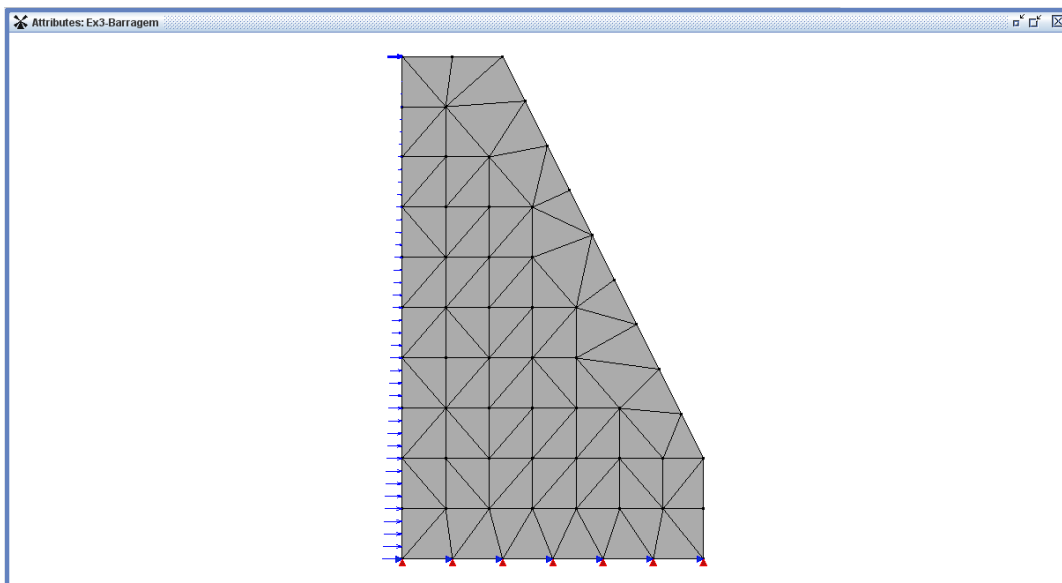
**Figura 4.14:** Barragem: Geometria.

No Módulo *Malha*, discretizou-se a geometria da estrutura, conforme apresentado na Figura 4.15, e definiram-se o modelo de análise e o tipo de elemento finito.



**Figura 4.15:** Barragem: Malha.

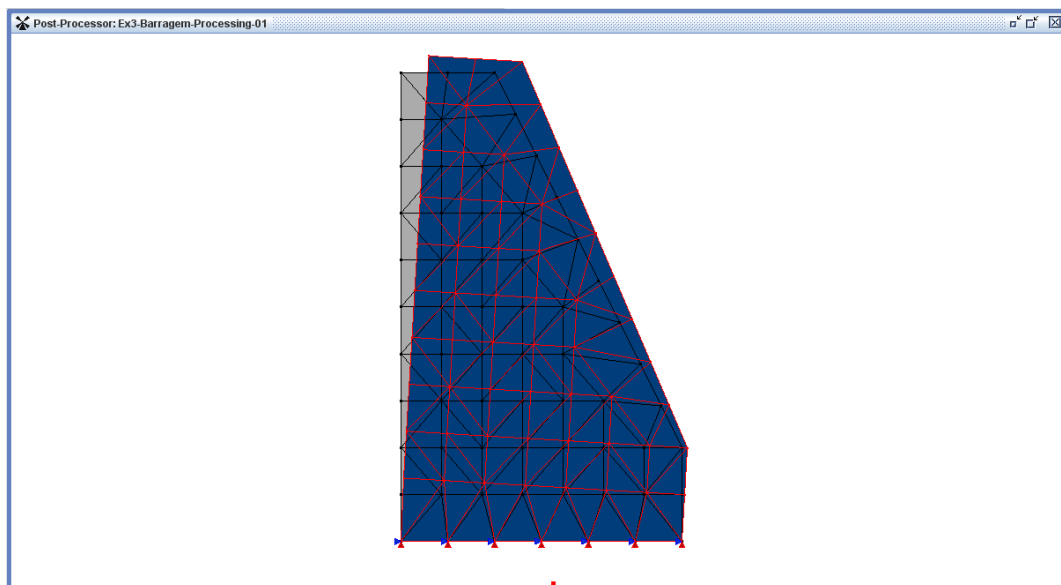
No Módulo *Atributos*, definiram-se as condições de contorno e o modelo constitutivo, criaram-se e atribuíram-se o material e a degeneração. Na Figura 4.16, apresenta-se a geometria da estrutura discretizada com as condições de contorno.



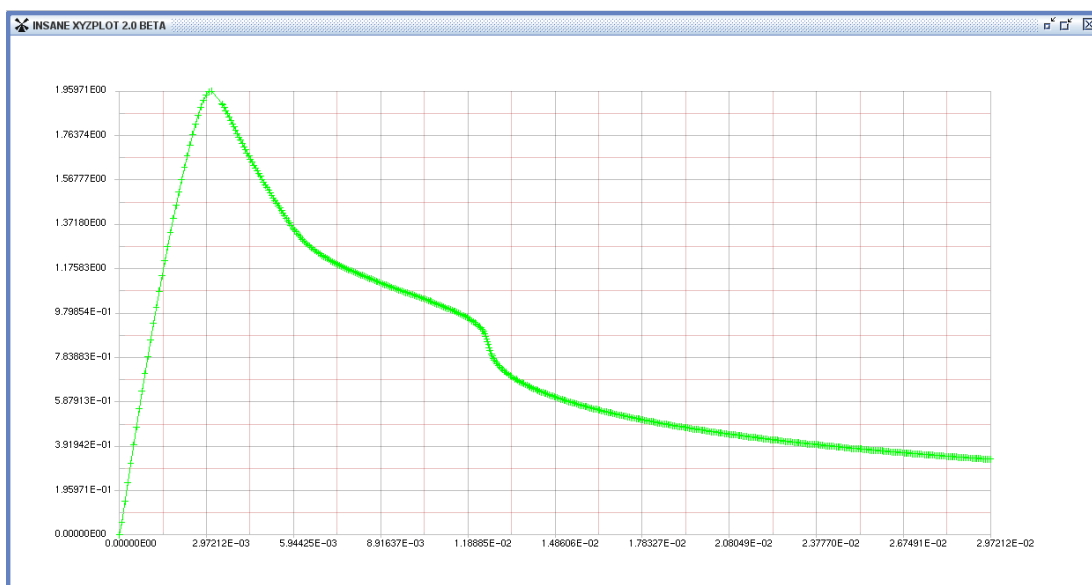
**Figura 4.16:** Barragem: Condições de contorno.



Na sequência, definiram-se os parâmetros da técnica de solução e processou-se a análise. No pós-processador, avaliaram-se a deformada no passo 500 ( $P = 0,33 MN$ ), apresentada na Figura 4.17 com fator de escala igual a 10, e a trajetória de equilíbrio correspondente ao deslocamento horizontal do ponto de aplicação da carga incremental, apresentada na Figura 4.18.



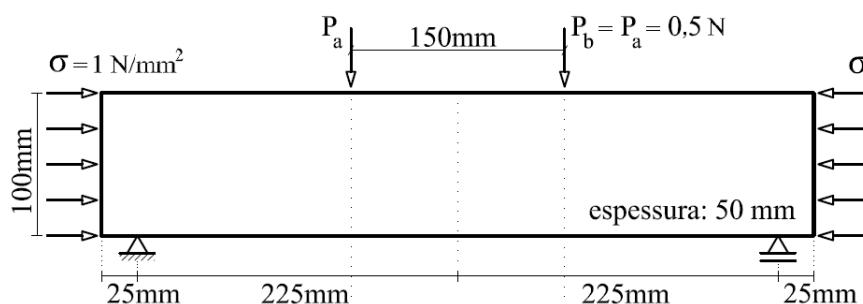
**Figura 4.17:** Barragem: Deformada no passo 500 ( $P = 0,33 MN$ ).



**Figura 4.18:** Barragem: Fator de carga  $\times$  Deslocamento horizontal do nó sob  $P$ .

## 4.4 Viga Pré-tensionada sob Flexão

A viga pré-tensionada sob flexão, cujas geometria, vinculações e cargas são ilustradas na Figura 4.19, foi simulada no sistema gráfico integrado objetivando-se destacar o emprego deste sistema para a opção de análise bidimensional estática fisicamente não linear com carregamento no plano pelo MEFG. Este exemplo foi adaptado de Penna (2011), que se baseou nos estudos de DeJong et al. (2008).



**Figura 4.19:** Viga pré-tensionada: Configuração (Penna, 2011).

Nesta simulação, diferentemente daquela em Penna (2011), enriqueceram-se os nós do modelo com função polinomial composta pelos monômios  $x$  e  $y$ . Sendo esta função incidente em todas as direções definidas pelos graus de liberdade nodal convencionais e acrescido um grau de liberdade nodal para cada monômio por direção, tiveram-se acrescidos 4 graus de liberdade por nó. Através deste enriquecimento, as funções interpoladoras lineares dos elementos finitos quadrilaterais de quatro nós ( $Q4$ ), empregados nesta análise, foram aprimoradas para funções quadráticas.

Para esta modelagem, adotaram-se os seguintes parâmetros:

- Opção de análise: Carregamento no plano
- Técnica de geração de malhas: Mapeamento Transfinito
- Modelo de análise: Estado Plano de Tensão
- Elementos finitos:  $Q4$
- Modelo constitutivo: Fissuras distribuídas com direção de fissuração fixa

- Materiais:

Material 1:

Tipo: Leis bilineares para tração e para compressão

Parâmetros:

Módulo de elasticidade elástico:  $E_0 = 32000,0 \text{ N/mm}^2$

Módulo tangente do ramo descendente:  $E_2 = 3000,0 \text{ N/mm}^2$

Coefficiente de Poisson:  $\nu = 0,2$

Tensão limite de resistência à compressão:  $f_c = 30,0 \text{ N/mm}^2$

Tensão limite de resistência à tração:  $f_t = 3,0 \text{ N/mm}^2$

Comprimento característico:  $h = 50 \text{ mm}$

Energia de fratura:  $G_f = 0,06 \text{ N/mm}$

Fator de retenção ao cisalhamento:  $\beta_r = 0,0$

- Degenerações:

Seção 1:

Tipo: Degeneração prescrita

Propriedades:

Espessura:  $50 \text{ mm}$

Material associado: Material 1

- Funções de enriquecimento nodal:

Enriquecimento 1:

Tipo: Polinomial

Parâmetros:

Número de monômios: 2

Expoentes  $p$  e  $q$  do 1<sup>o</sup> monômio  $x^p y^q$ : 1 0

Expoentes  $p$  e  $q$  do 2<sup>o</sup> monômio  $x^p y^q$ : 0 1

- Carregamentos:

Não incremental:  $\sigma = 1,0 \text{ N/mm}^2$  aplicada como  $f_x = 50,0 \text{ N/mm}$

Incremental:  $P_a = P_b = 0,5 \text{ N}$

- Parâmetros da técnica de solução:

Processo incremental-iterativo: Newton-Raphson Padrão

Método de controle: Deslocamento generalizado

Fator de carga inicial: 10,0

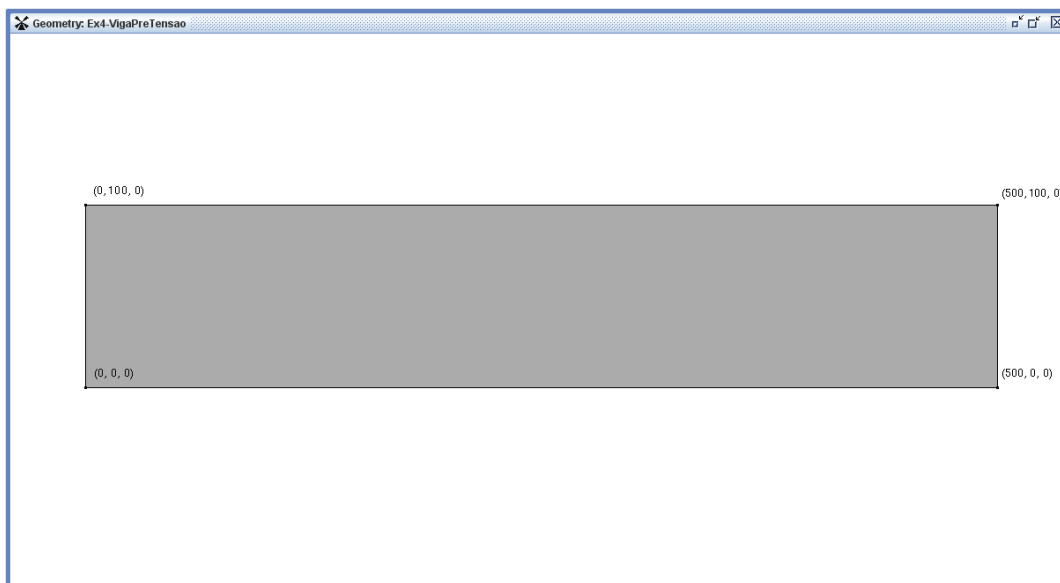
Número máximo de passos: 1500

Número máximo de iterações: 200

Tolerância:  $1,0 \times 10^{-4}$

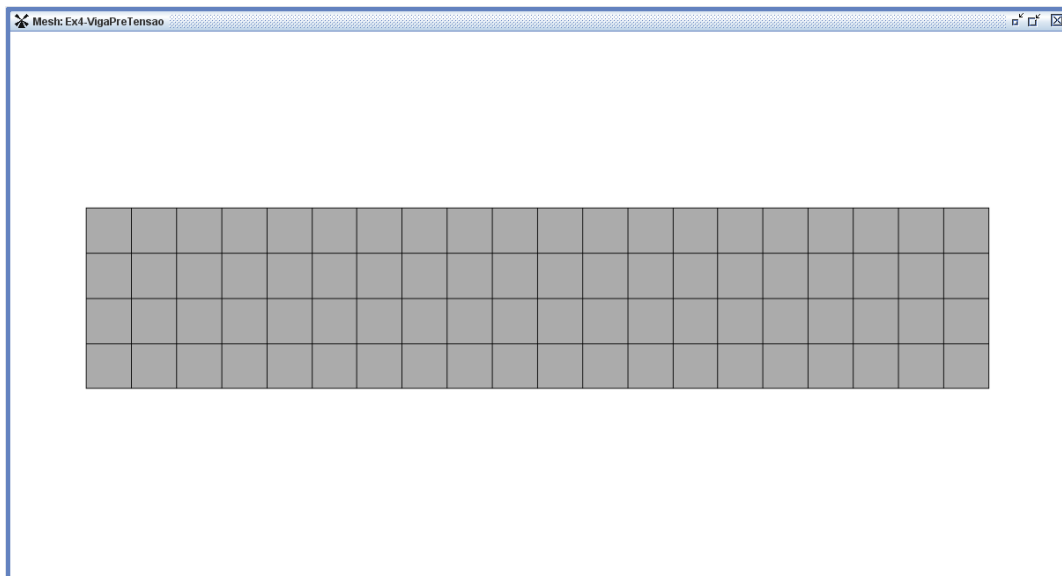
Tipo de convergência: Magnitude do vetor de deslocamentos incrementais

No Módulo *Geometria*, desenhou-se a geometria da estrutura, conforme apresentado na Figura 4.20.



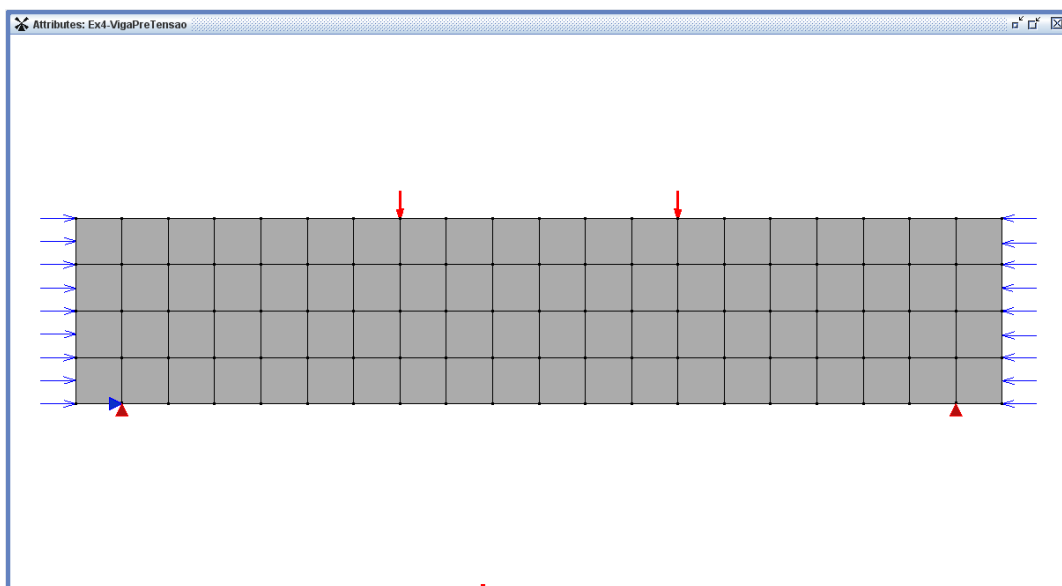
**Figura 4.20:** Viga pré-tensionada: Geometria.

No Módulo *Malha*, discretizou-se a geometria da estrutura, conforme apresentado na Figura 4.21, e definiram-se o modelo de análise e o tipo de elemento finito.



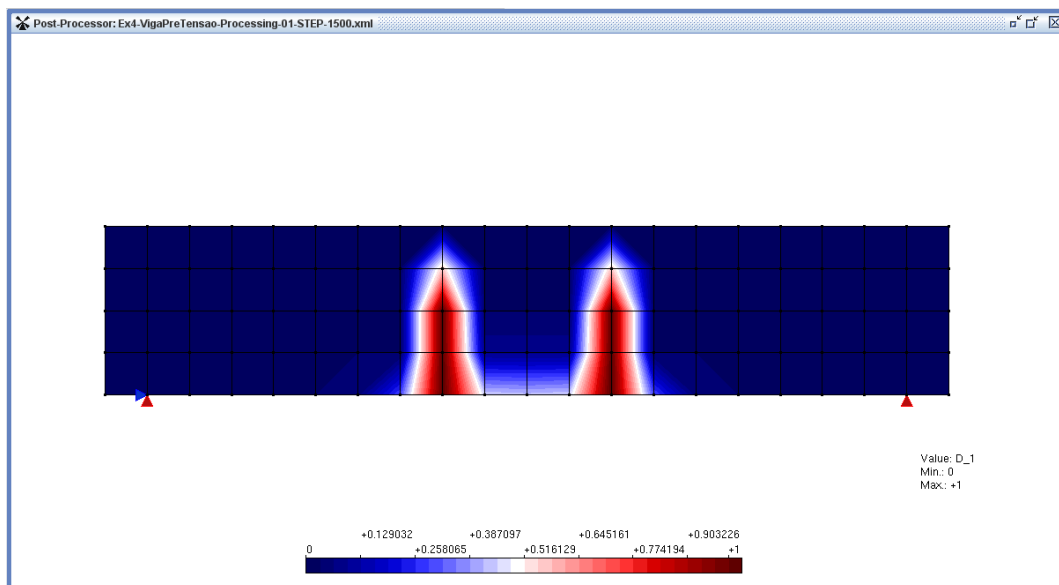
**Figura 4.21:** Viga pré-tensionada: Malha.

No Módulo *Atributos*, definiram-se as condições de contorno e o modelo constitutivo, criaram-se e atribuíram-se o material, a degeneração e a função de enriquecimento. Na Figura 4.22, apresenta-se a geometria da estrutura discretizada com as condições de contorno.

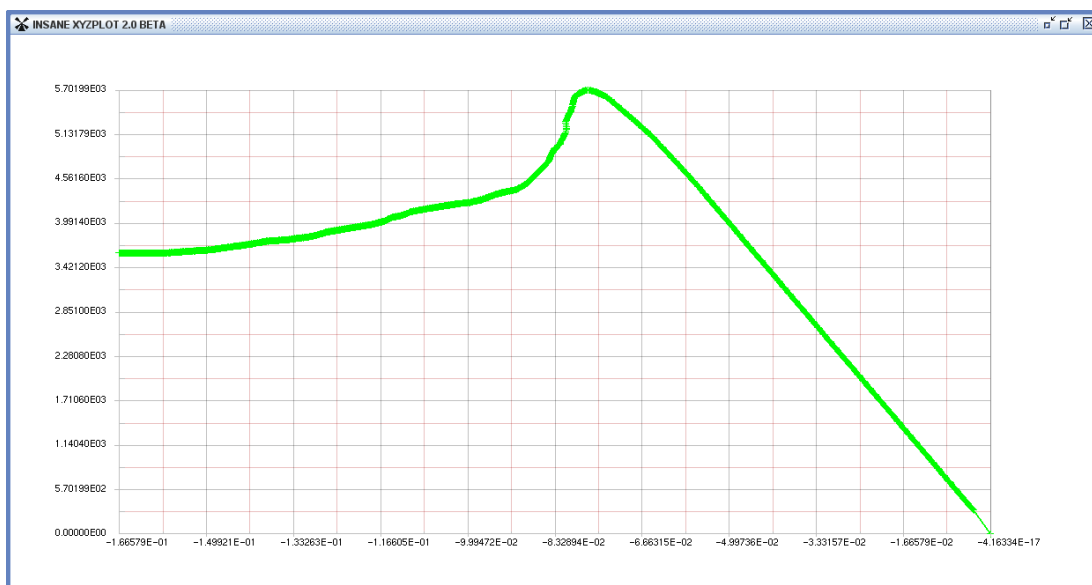


**Figura 4.22:** Viga pré-tensionada: Condições de contorno.

Na sequência, definiram-se os parâmetros da técnica de solução e processou-se a análise. No pós-processador, avaliaram-se a variação do dano na direção de  $\varepsilon_1$  ao longo do domínio no passo 1500 ( $P_a = 1,81 \text{ kN}$ ), apresentada na Figura 4.23, e a trajetória de equilíbrio correspondente ao deslocamento vertical do ponto de aplicação da carga  $P_a$ , apresentada na Figura 4.24.



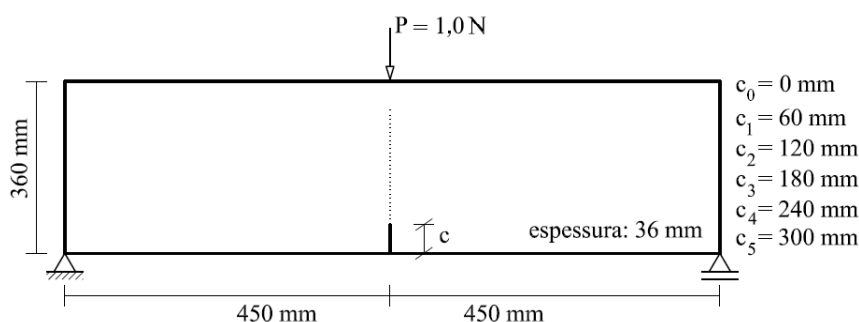
**Figura 4.23:** Viga pré-tensionada: Dano na direção de  $\varepsilon_1$  no passo 1500 ( $P_a = 1,81 \text{ kN}$ ).



**Figura 4.24:** Viga pré-tensionada: Fator de carga  $\times$  Deslocamento vertical do nó sob  $P_a$ .

## 4.5 Viga com Pré-trinca sob Flexão

A viga com pré-trinca sob flexão, cujas geometria, vinculações e carga são ilustradas na Figura 4.25, foi reproduzida de Penna (2011) no sistema gráfico integrado objetivando-se demonstrar a capacidade do recurso de modelagem de descontinuidades preexistentes na estrutura, aplicável à opção de análise com carregamento no plano. Por serem simuladas por meio de funções de enriquecimento nodal, a existência destas descontinuidades no modelo condiciona a análise ao MEFG.



**Figura 4.25:** Viga com pré-trinca: Configuração (Penna, 2011).

Conforme em Penna (2011), através deste exemplo, avaliou-se a ductilidade estrutural de uma viga submetida à flexão em três pontos. A partir do crescimento de uma trinca inicial, verificou-se a repercussão da variação da altura efetiva da viga no seu comportamento estrutural. As dimensões da viga foram mantidas constantes e o comprimento da trinca foi aumentado em cada análise.

Diferentemente da simulação em Penna (2011) cujo crescimento da trinca inicial foi inserido no modelo de forma discreta pela duplicação de nós, nesta simulação, os diversos estágios da trinca foram modelados através do recurso que permite considerá-la sem a necessidade de introduzi-la na geometria do modelo. A partir deste recurso, as descontinuidades preexistentes são simuladas por meio de funções degrau, que consistem em funções de enriquecimento nodal, porém, criadas e atribuídas aos nós necessários pelo programa, cabendo ao analista apenas a definição dos pontos inicial e final destas descontinuidades.

Para esta modelagem, adotaram-se os seguintes parâmetros:

- Opção de análise: Carregamento no plano
- Técnica de geração de malhas: Mapeamento Transfinito
- Modelo de análise: Estado Plano de Tensão
- Elementos finitos: Q4
- Modelo constitutivo: Fissuras distribuídas com direção de fissuração fixa
- Materiais:

Material 1:

Tipo: Carreira e Chu (1985), compressão, Boone et al. (1986), tração

Parâmetros:

Módulo de elasticidade elástico:  $E_0 = 44000,0 \text{ N/mm}^2$

Coefficiente de Poisson:  $\nu = 0,2$

Tensão limite de resistência à compressão:  $f_c = 40,0 \text{ N/mm}^2$

Tensão limite de resistência à tração:  $f_t = 3,8 \text{ N/mm}^2$

Deformação relativa ao limite elástico na compressão:  $\varepsilon_c = 0,0018$

Energia de fratura:  $G_f = 0,164 \text{ N/mm}$

Comprimento característico:  $h = 100 \text{ mm}$

Fator de retenção ao cisalhamento:  $\beta_r = 0,05$

- Degenerações:

Seção 1:

Tipo: Degeneração prescrita

Propriedades:

Espessura:  $36 \text{ mm}$

Material associado: Material 1



- Descontinuidades (entalhes):

Descontinuidade 1:

Tipo: Simulada pelo MEFG

Parâmetros:

1<sup>a</sup> Análise ( $C_0$ ): Sem trinca

2<sup>a</sup> Análise ( $C_1$ ):  $\left\{ \begin{array}{l} \text{Ponto inicial } (x \ y): (450, 0 \ 0, 0) \\ \text{Ponto final } (x \ y): (450, 0 \ 60, 0) \end{array} \right.$

3<sup>a</sup> Análise ( $C_2$ ):  $\left\{ \begin{array}{l} \text{Ponto inicial } (x \ y): (450, 0 \ 0, 0) \\ \text{Ponto final } (x \ y): (450, 0 \ 120, 0) \end{array} \right.$

4<sup>a</sup> Análise ( $C_3$ ):  $\left\{ \begin{array}{l} \text{Ponto inicial } (x \ y): (450, 0 \ 0, 0) \\ \text{Ponto final } (x \ y): (450, 0 \ 180, 0) \end{array} \right.$

5<sup>a</sup> Análise ( $C_4$ ):  $\left\{ \begin{array}{l} \text{Ponto inicial } (x \ y): (450, 0 \ 0, 0) \\ \text{Ponto final } (x \ y): (450, 0 \ 240, 0) \end{array} \right.$

6<sup>a</sup> Análise ( $C_5$ ):  $\left\{ \begin{array}{l} \text{Ponto inicial } (x \ y): (450, 0 \ 0, 0) \\ \text{Ponto final } (x \ y): (450, 0 \ 300, 0) \end{array} \right.$

- Carregamentos:

Incremental:  $P = 1, 0 \ N$

- Parâmetros da técnica de solução:

Processo incremental-iterativo: Newton-Raphson Padrão

Método de controle: Comprimento de arco cilíndrico

Fator de carga inicial: 500, 0

Número máximo de passos: 200

Número máximo de iterações: 100

Tolerância:  $1, 0 \times 10^{-4}$

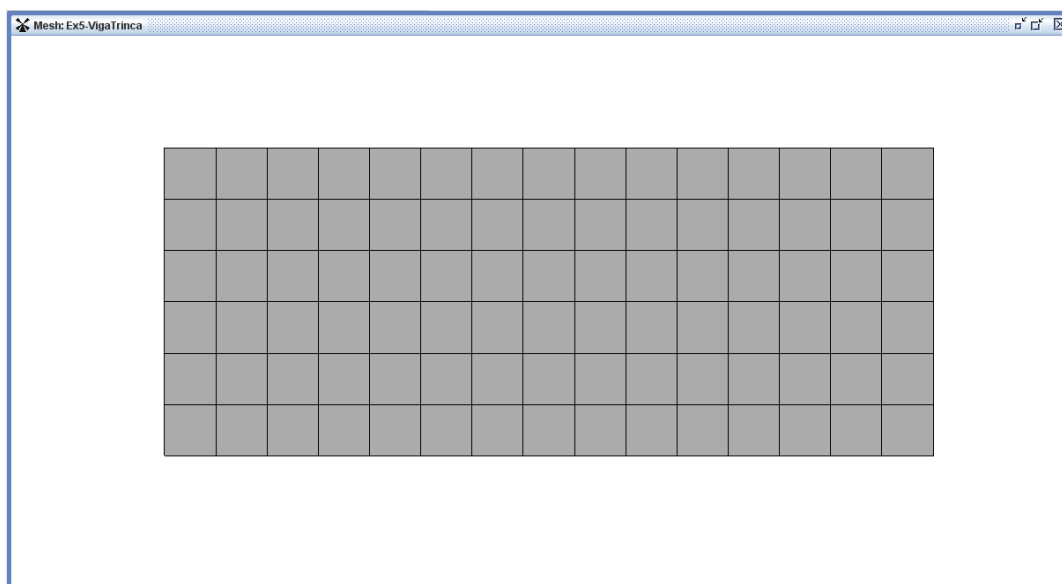
Tipo de convergência: Magnitude do vetor de deslocamentos incrementais

No Módulo *Geometria*, desenhou-se a geometria da estrutura, conforme apresentado na Figura 4.26.



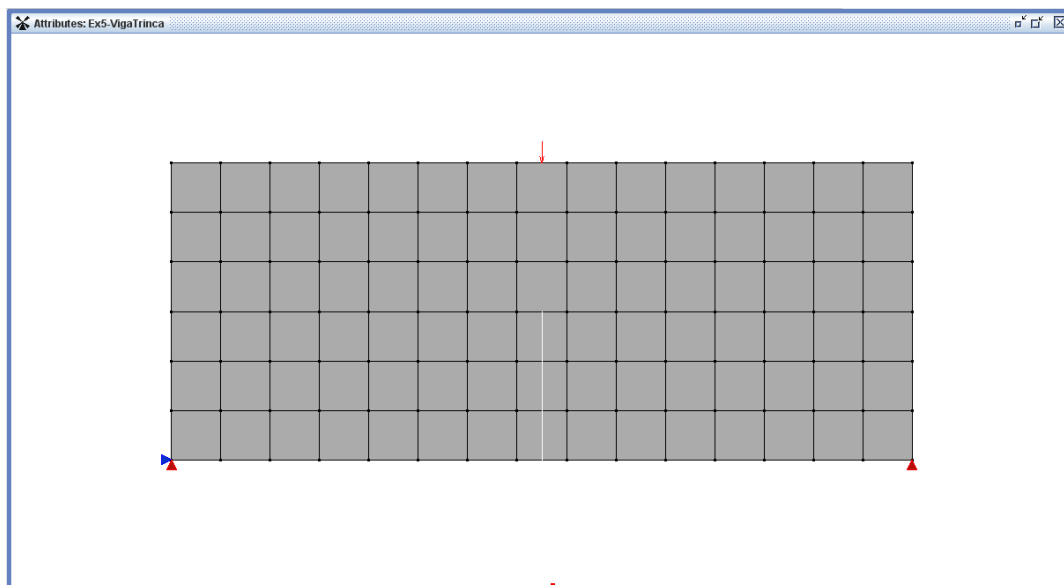
**Figura 4.26:** Viga com pré-trinca: Geometria.

No Módulo *Malha*, discretizou-se a geometria da estrutura, conforme apresentado na Figura 4.27, e definiram-se o modelo de análise e o tipo de elemento finito.



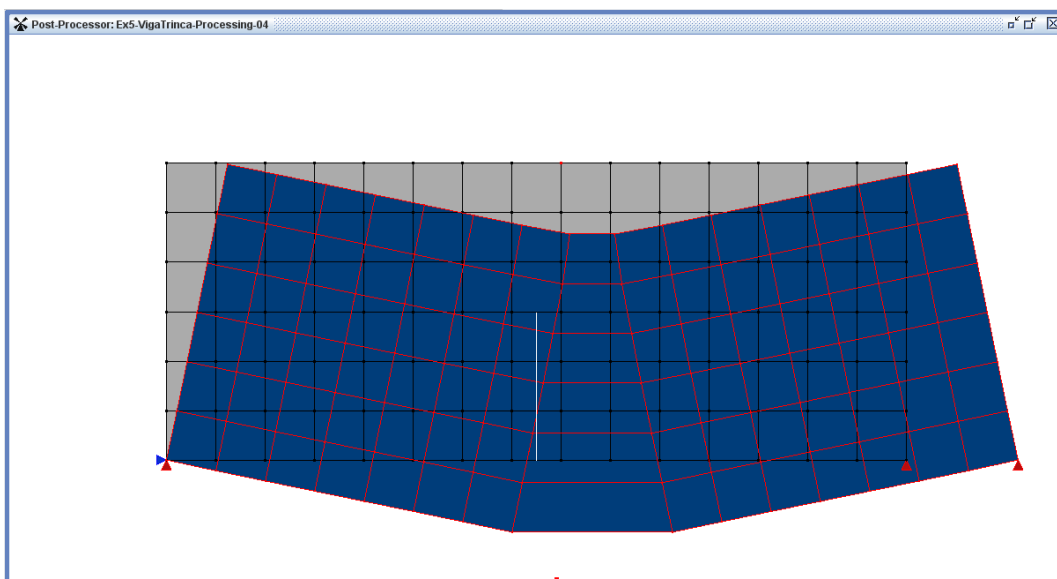
**Figura 4.27:** Viga com pré-trinca: Malha.

No Módulo *Atributos*, definiram-se as condições de contorno e o modelo constitutivo, criaram-se e atribuíram-se o material e a degeneração e modelou-se em cada análise a descontinuidade correspondente. Na Figura 4.28, apresenta-se a geometria da estrutura discretizada com as condições de contorno, destacando-se a descontinuidade  $C_3$ , correspondente à 4ª análise.

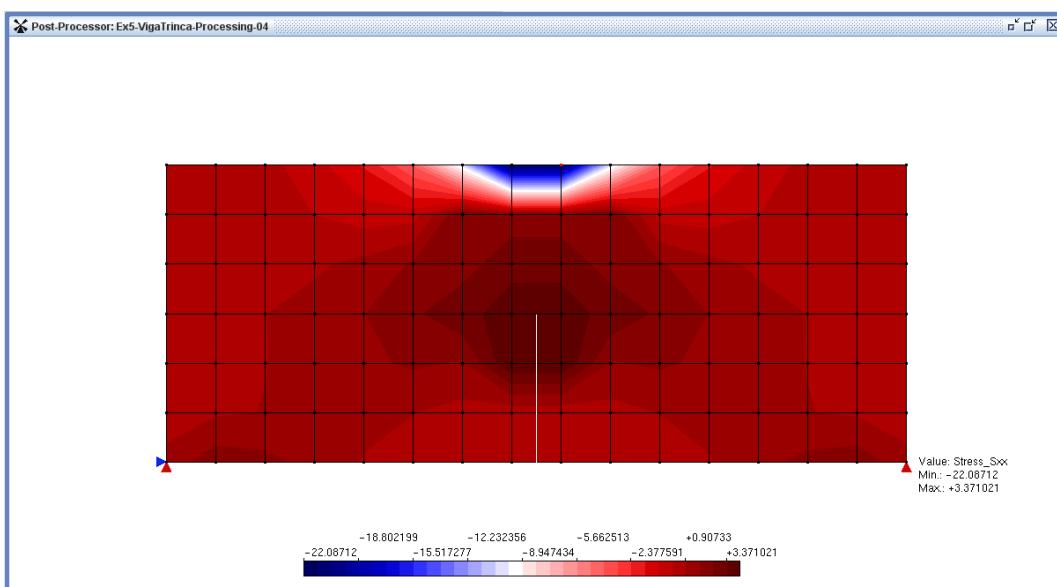


**Figura 4.28:** Viga com pré-trinca: Condições de contorno.

Na sequência, definiram-se os parâmetros da técnica de solução e processou-se a análise. No pós-processador, avaliaram-se o salto no campo de deslocamentos, ilustrado na Figura 4.29 por meio da deformada da 4ª análise ( $C_3$ ) no passo 200 ( $P = 7,4 \text{ kN}$ ), apresentada com fator de escala igual a 100, e a concentração de tensões na região próxima à ponta da trinca, ilustrada na Figura 4.30 através da variação da tensão  $\sigma_{xx}$  ao longo do domínio naquele mesmo passo, daquela mesma análise. A partir da avaliação do salto no campo de deslocamentos e da concentração de tensões na região próxima à ponta da trinca, destaca-se a capacidade do recurso de modelagem de descontinuidades preexistentes na estrutura simular os efeitos da presença de pré-trincas sem a necessidade de introduzi-las na geometria do modelo.

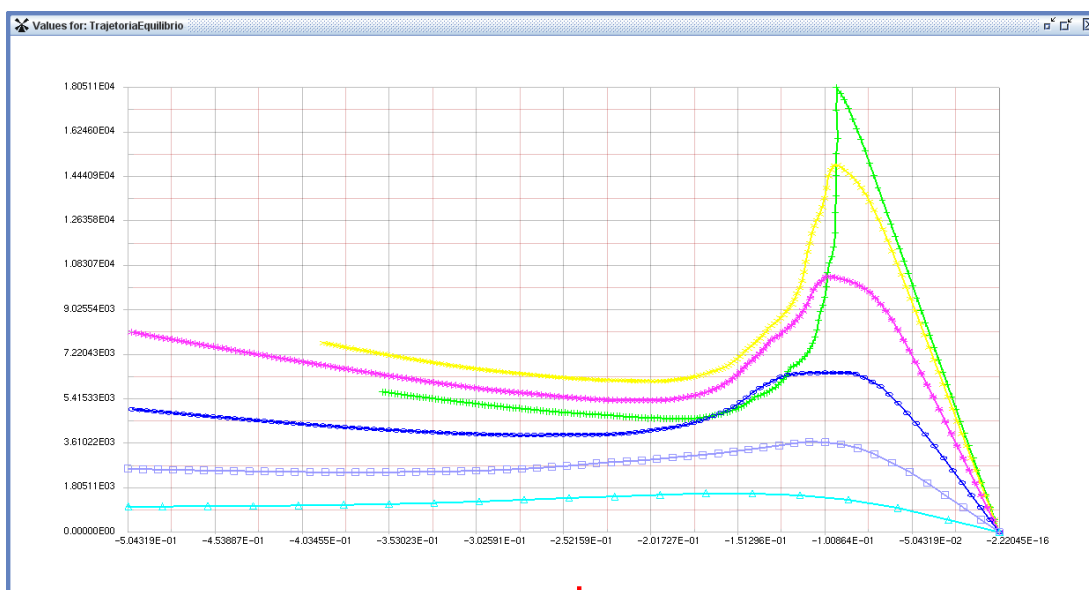


**Figura 4.29:** Viga com pré-trinca: Deformada no passo 200 ( $P = 7,4 \text{ kN}$ ).



**Figura 4.30:** Viga com pré-trinca: Tensão  $\sigma_{xx}$  no passo 200 ( $P = 7,4 \text{ kN}$ ).

Na Figura 4.31, apresentam-se as trajetórias de equilíbrio correspondentes ao deslocamento vertical do ponto de aplicação da carga  $P$ . As análises podem ser identificadas pelo valor da carga máxima, que reduziu à medida que se aumentou o comprimento da trinca. Através deste comparativo, a repercussão da variação da altura efetiva da viga no seu comportamento estrutural pôde ser avaliada.



**Figura 4.31:** Viga com pré-trinca: Fator de carga  $\times$  Deslocamento vertical sob  $P$ .

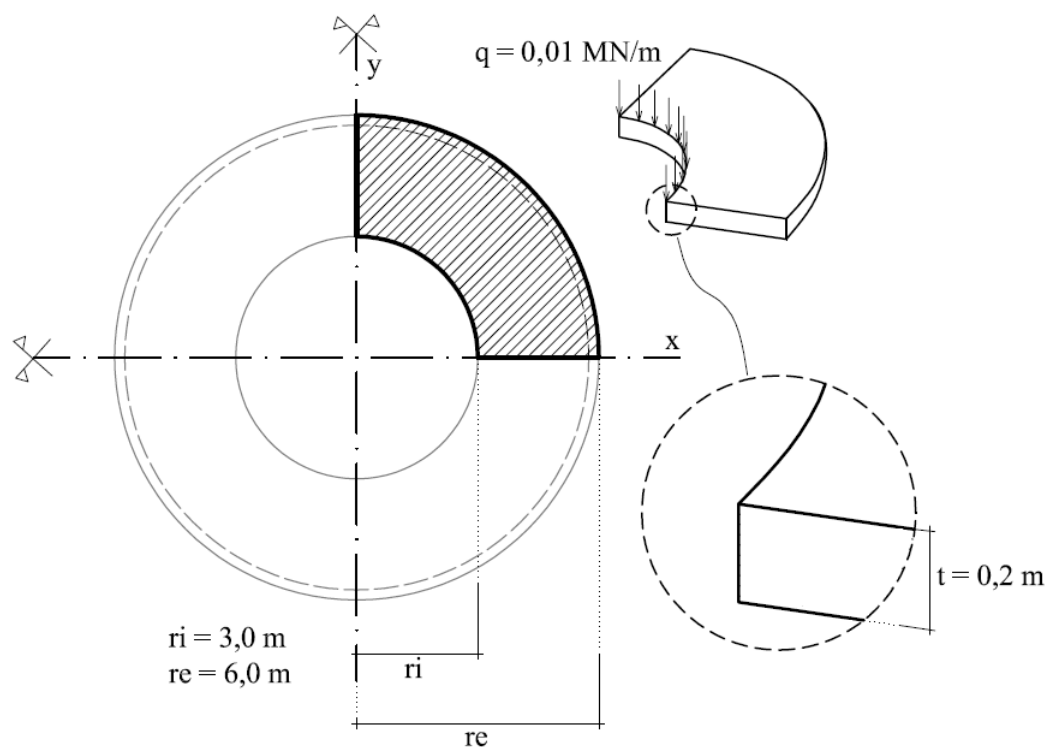
Observa-se que, quanto maior a altura efetiva da viga, mais acentuado é o ramo descendente da trajetória de equilíbrio e menor é o valor em módulo do deslocamento associado à carga máxima. Desta forma, verifica-se a conformidade qualitativa dos resultados em termos da ductilidade estrutural da viga, que aumenta com o crescimento da trinca.

Em Penna (2011), avaliou-se ainda a variação da resistência da viga com a variação de sua altura efetiva. A resistência nominal da viga foi calculada a partir da definição da resistência de flexão proposta por Planas et al. (1999), para os valores de carga máxima. Observou-se uma queda da resistência da viga com o aumento de sua altura efetiva.

Conforme em Penna (2011), observa-se também que, à medida em que a trinca inicial cresce, o comportamento se assemelha ao efeito da diminuição da altura da viga em um estudo do efeito de tamanho. Logo, os resultados aqui apresentados mostram que é possível ainda simular, qualitativamente, o efeito de tamanho, sem a necessidade de redefinição do modelo para diferentes tamanhos estruturais.

## 4.6 Placa Anular com Carga na Borda Interna

A placa anular com carga na borda interna, cujas geometria, vinculação e carga são ilustradas na Figura 4.32, foi simulada no sistema gráfico integrado objetivando-se destacar o emprego deste sistema para a opção de análise bidimensional estática linear com carregamento normal ao plano (placas) pelo MEF.



**Figura 4.32:** Placa anular: Configuração (Adaptado de Penna (2011)).

Para esta modelagem, adotaram-se os seguintes parâmetros:

- Opção de análise: Carregamento normal ao plano (placas)
- Técnica de geração de malhas: Mapeamento Transfinito
- Modelo de análise: Placa de Reissner Mindlin
- Elementos finitos: Q8
- Modelo constitutivo: Linear elástico

- Materiais:

Material 1:

Tipo: Linear elástico isotrópico

Parâmetros:

Módulo de elasticidade:  $E = 28000,0 \text{ MPa}$

Coefficiente de Poisson:  $\nu = 0,2$

- Degenerações:

Seção 1:

Tipo: Degeneração prescrita

Propriedades:

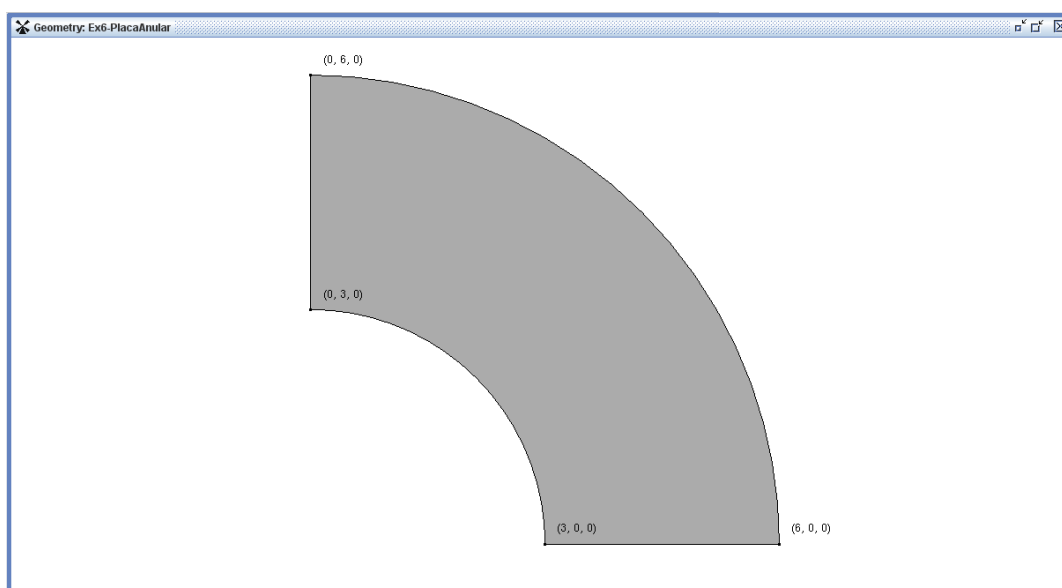
Espessura:  $0,2 \text{ m}$

Material associado: Material 1

- Carregamentos:

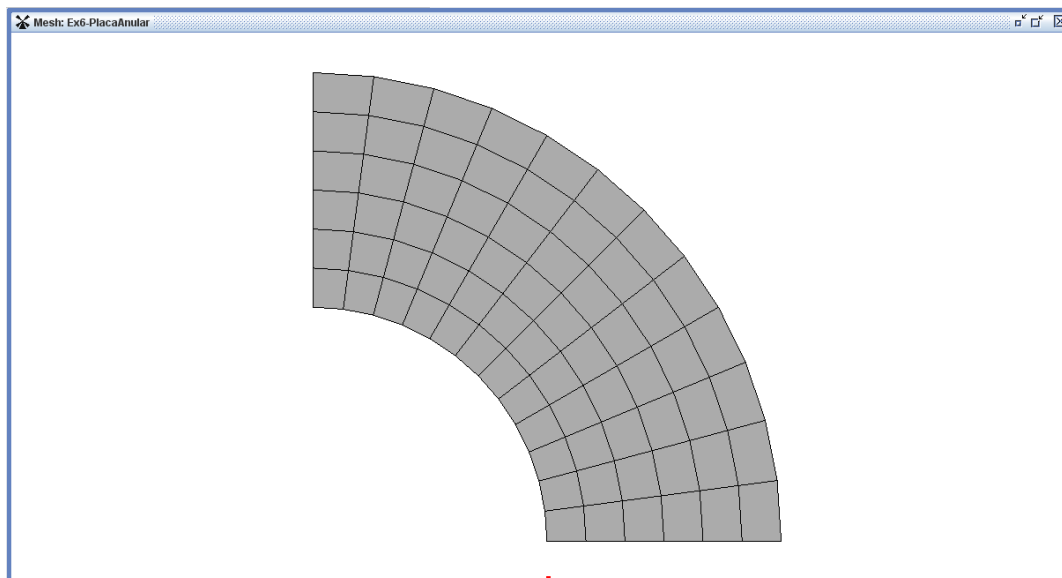
Não incremental:  $q = 0,01 \text{ MN/m}$

No Módulo *Geometria*, desenhou-se a geometria da estrutura, conforme apresentado na Figura 4.33.



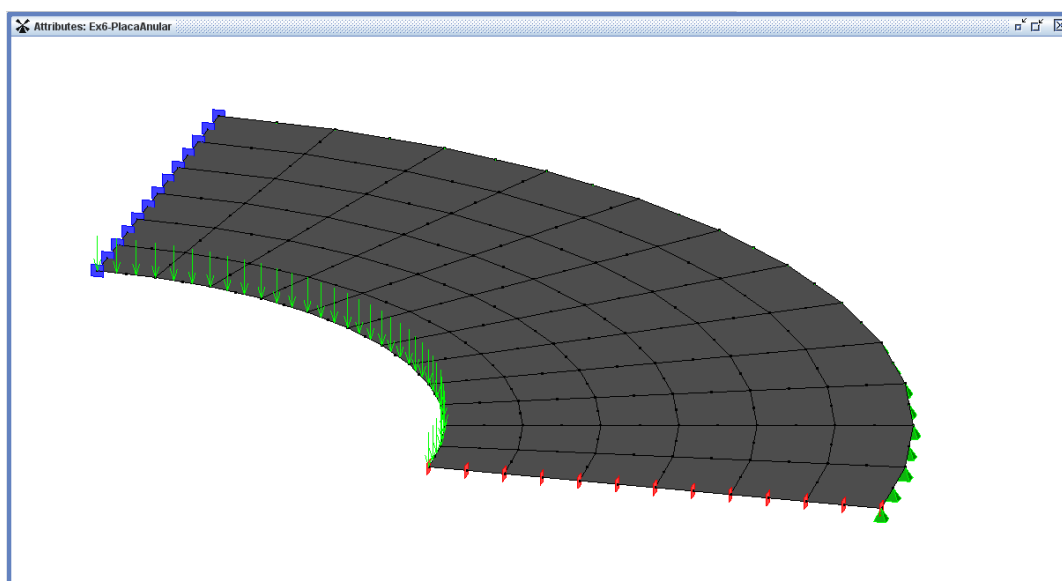
**Figura 4.33:** Placa anular: Geometria.

No Módulo *Malha*, discretizou-se a geometria da estrutura, conforme apresentado na Figura 4.34, e definiram-se o modelo de análise e o tipo de elemento finito.



**Figura 4.34:** Placa anular: Malha.

No Módulo *Atributos*, definiram-se as condições de contorno, criaram-se e atribuíram-se o material e a degeneração. Ressalta-se que, nesta opção de análise, não se define o modelo constitutivo, condicionado ao linear elástico. Na Figura 4.35, apresenta-se a geometria da estrutura discretizada com as condições de contorno.



**Figura 4.35:** Placa anular: Condições de contorno.



No pós-processador, avaliaram-se a deformada, apresentada na Figura 4.36 com fator de escala igual a 20, e a variação da flecha ao longo do domínio, apresentada na Figura 4.37.

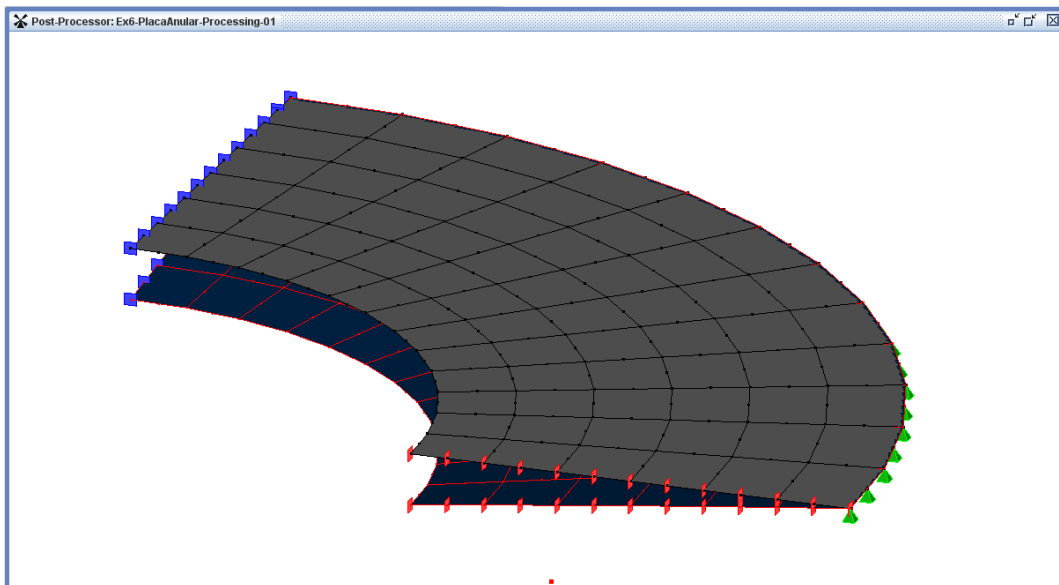


Figura 4.36: Placa anular: Deformada.

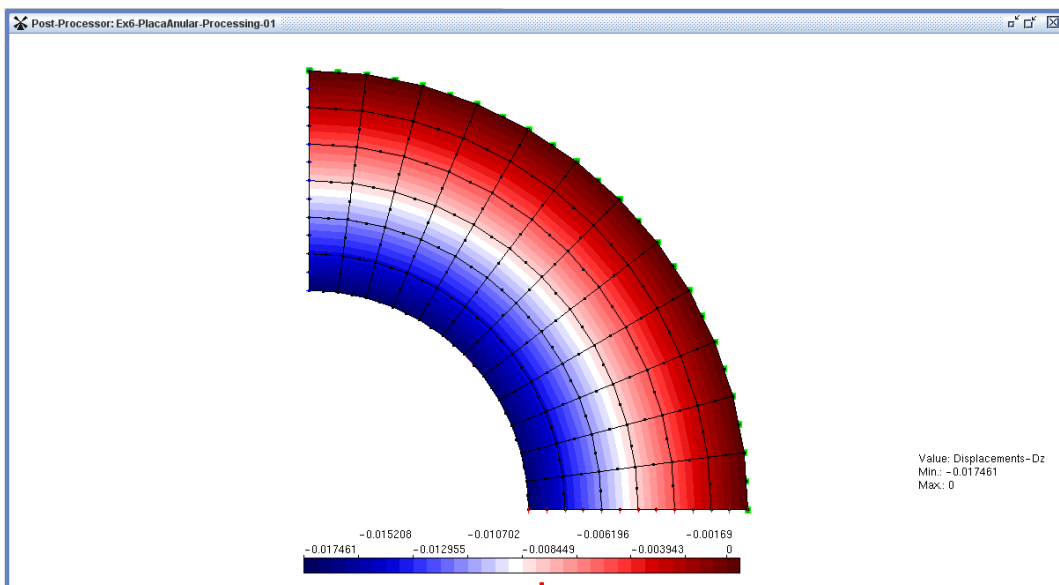


Figura 4.37: Placa anular: Deslocamento  $D_z$ .

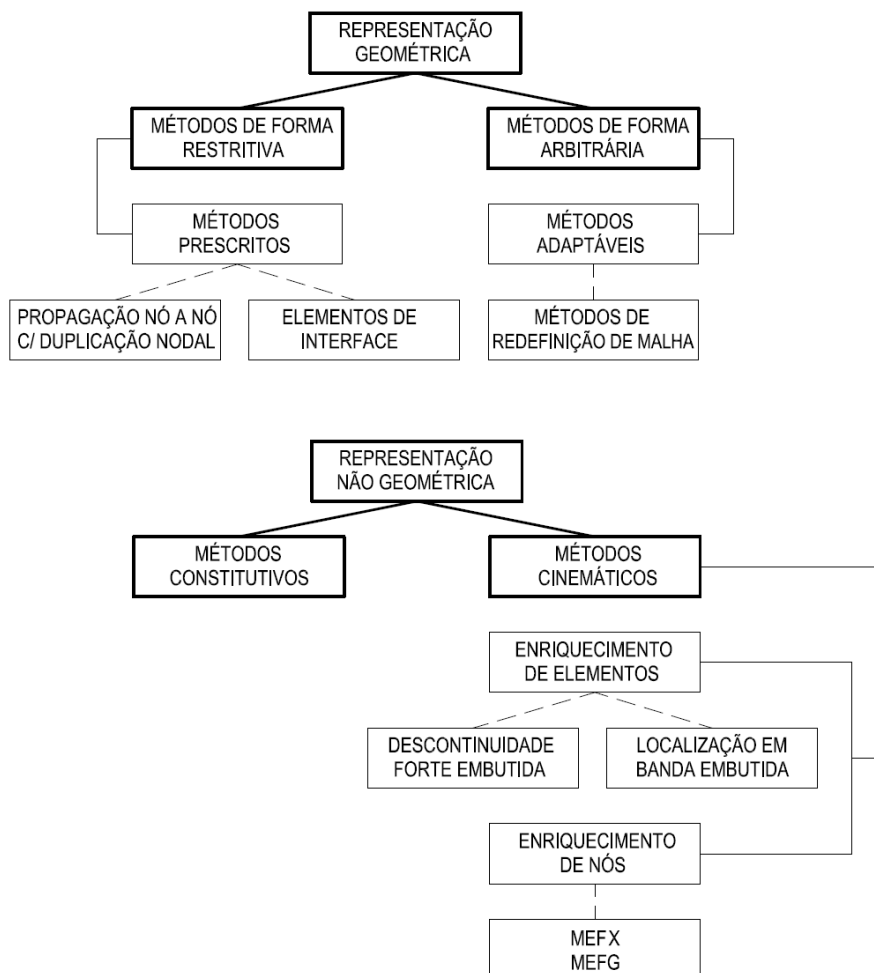
## Capítulo 5

# MÉTODOS NUMÉRICOS PARA FISSURAÇÃO

Neste capítulo, abordam-se os principais métodos numéricos baseados no MEF empregados na simulação dos modelos físicos de fratura. Ressalva-se a existência de vários outros métodos capazes de tratar numericamente os modelos de fratura, tais como aqueles baseados no Método das Diferenças Finitas, nos Métodos Sem Malha e no Método dos Elementos de Contorno.

Em Ingraffea e Wawrzynek (2004), apresenta-se uma classificação dos principais métodos numéricos usados em diversas aplicações da Mecânica da Fratura. Nesta classificação, dividem-se os diversos métodos numéricos em dois grupos principais, que se distinguem pela maneira como representam a propagação da fissuração. O primeiro grupo representa a propagação da fissuração de forma *geométrica* e o segundo de forma *não geométrica*. Assim, nos modelos numéricos do primeiro grupo, a fissura é uma entidade geométrica inserida na geometria do modelo e tanto a geometria quanto a discretização do modelo, se necessário, mudam conforme a propagação da fissuração. Já nos modelos numéricos do segundo grupo, a propagação da fissuração é representada dentro do modelo constitutivo do material ou através de um modelo cinemático, no qual a fissura é representada por meio de saltos ou descontinuidades nos campos de deslocamentos e/ou de deformações. Assim, neste segundo grupo, nem a geometria do modelo nem a sua discretização contém a fissura ou precisa sofrer alterações durante a sua propagação (Wolff, 2010).

A partir desta classificação proposta por Ingraffea e Wawrzynek (2004), tem-se a classificação dos métodos numéricos para fissuração que se baseiam no MEF, reproduzida de Wolff (2010), na Figura 5.1.



**Figura 5.1:** Classificação dos métodos numéricos para fissuração que se baseiam no MEF (Adaptado de Ingraffea e Wawrzynek (2004)).

## 5.1 Métodos de Representação Geométrica

Conforme Ingraffea e Wawrzynek (2004), os Métodos de Representação Geométrica podem ser divididos em dois grupos: aqueles que representam a fissuração de forma geométrica-restritiva e aqueles que representam a fissuração de forma geométrica-arbitrária.

Entre os Métodos de Forma Restritiva, têm-se os Métodos Prescritos, nos quais a disposição das fissuras no modelo é restrita aos contornos dos elementos. Esta restrição condiciona a geometria da fissura ao tamanho dos elementos e o caminho de sua propagação à malha, tornando-se necessário o conhecimento prévio deste caminho para que a discretização seja convenientemente ajustada. Estes métodos prescrevem a propagação da fissuração limitando-a à discretização.

Quanto à simulação da fissura, os Métodos de Forma Restritiva se dividem em Propagação Nó a Nó com Duplicação Nodal e o emprego de Elementos de Interface.

Segundo Wolff (2010), os métodos de Propagação Nó a Nó com Duplicação Nodal foram os primeiros a serem usados em estudos numéricos baseados no MEF para simular o processo de fissuração. Nestes métodos, os nós são duplicados e separados à medida em que a fissura se propaga entre os elementos, resultando na alteração da malha, porém, sem modificar a geometria dos elementos.

Conforme Ingraffea e Wawrzynek (2004), na época em que estes métodos começaram a ser usados, haviam alguns limitantes: a capacidade e a velocidade de processamento dos computadores eram bem limitadas e o aumento do tamanho das matrizes de rigidez provocado pela duplicação dos nós dificultava ainda mais o processamento da análise. Estas limitações vêm sendo contornadas com o aumento da capacidade de memória e de processamento dos computadores, aliado ao desenvolvimento de rotinas para minimizar a largura de banda e à novas estratégias de soluções iterativas. Em Ngo e Scordelis (1967), encontra-se um exemplo de aplicação dos métodos de Propagação Nó a Nó com Duplicação Nodal.

Uma versão derivada dos métodos de Propagação Nó a Nó com Duplicação Nodal, é o emprego de Elementos de Interface. Para tanto, inserem-se os Elementos de Interface entre os elementos-padrão da malha, ajustando-os aos contornos destes elementos-padrão. Desta forma, o caminho da fissura pode ser determinado a partir de uma análise prévia dos prováveis caminhos por onde a fissura possa passar, dispondo os Elementos de Interface naqueles que melhor se encaixam à solução.

Outra vantagem dos Elementos de Interface é a possibilidade de empregá-los em um modelo não linear de fissura coesiva, por exemplo, para separar gradualmente os contornos dos elementos-padrão. Em López et al. (2008), encontra-se um exemplo do emprego de Elementos de Interface na simulação numérica do comportamento de materiais parcialmente frágeis.

Entre os Métodos de Forma Arbitrária, têm-se os Métodos Adaptáveis, nos quais o modelo é modificado para se adaptar à geometria da fissura à medida em que esta se propaga. Os Métodos de Redefinição de Malha pertencem a esta categoria e permitem que uma fissura seja modelada de forma completamente arbitrária dentro de um modelo, uma vez que a geometria do modelo e a sua discretização são atualizadas para se ajustarem à geometria da fissura durante a sua evolução.

Conforme Wolff (2010), os Métodos de Redefinição de Malha usam as ferramentas mais modernas que estão disponíveis no MEF para a resolução dos campos de deslocamentos, de deformações e de tensões, aliadas ao uso de avançadas tecnologias para geração automática de malhas e para o mapeamento das informações do estado da discretização antes e depois de cada redefinição da malha.

Segundo Ingraffea e Wawrzynek (2004), o componente fundamental destes métodos é o banco de dados topológico, que organiza e controla a troca de informações entre a geometria do modelo e cada configuração da malha que representa a estrutura fissurada. Devido à geração de diversas malhas durante a análise, uma descrição geométrica do modelo deve ser mantida e atualizada como parte do processo de simulação de forma independente das discretizações numéricas. Desta forma, outro componente fundamental é o banco de dados geométrico, que contém uma descrição explícita do modelo e da fissura. Em Anderson et al. (2005), apresenta-se um algoritmo de redefinição de malha, ao passo que em Zheng et al. (2005), apresenta-se uma aplicação deste algoritmo no estudo da interface de interação de fluidos. Em Dao et al. (2010), encontra-se um exemplo de aplicação dos Métodos de Redefinição de Malha para a simulação da corrosão do aço em estruturas de concreto armado.

## 5.2 Métodos de Representação Não Geométrica

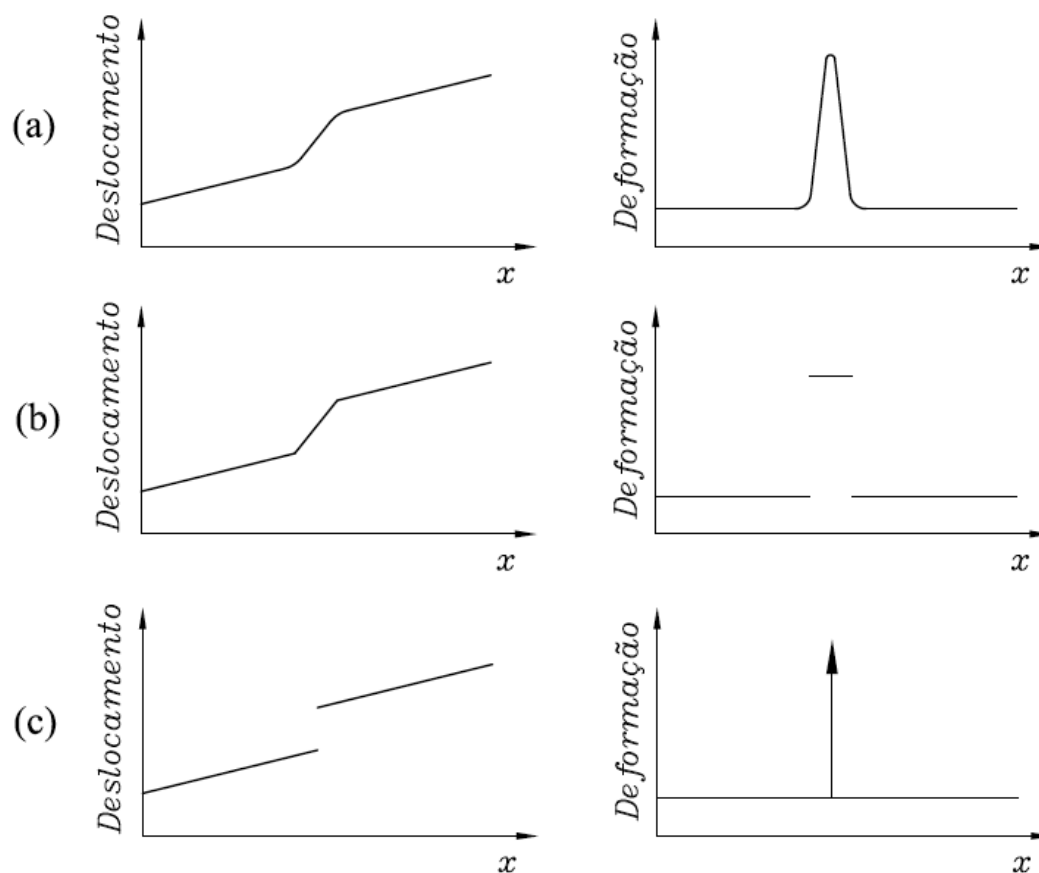
Conforme Ingraffea e Wawrzynek (2004), os Métodos de Representação Não Geométrica podem ser divididos em dois grupos: aqueles que representam a fissuração no modelo constitutivo e aqueles que representam a propagação da fissura através dos campos cinemáticos. O principal objetivo destes métodos é modelar a fissuração sem a necessidade de alterações na geometria do modelo e em sua discretização.

Nos Métodos Constitutivos, a rigidez do material em estudo sofre uma degradação na região fissurada, de forma a simular a descontinuidade do campo de deslocamentos provocada pela fissuração, ao passo que nos Métodos Cinemáticos, o efeito da fissuração nos campos de deslocamentos e de deformações é embutido dentro dos elementos, seja pelo Enriquecimento de Elementos, de forma local, ou pelo Enriquecimento de Nós, de forma global.

Estes enriquecimentos consistem em funções degrau (funções descontínuas) introduzidas nas equações de interpolação dos campos de deslocamentos e/ou de deformações dos elementos. A partir destes enriquecimentos, incorporam-se saltos ou descontinuidades nos campos cinemáticos na zona de processo de fratura, permitindo-se a representação do comportamento da fissuração.

Conforme Wolff (2010), no grupo de Enriquecimento de Elementos, aplica-se uma combinação linear da função degrau com as funções de forma padrão do elemento enriquecido mais as funções de forma padrão dos demais elementos da vizinhança que estejam associadas aos nós separados pela descontinuidade, de tal maneira que a função resultante anula-se em todos os nós. Este tipo de enriquecimento cria a possibilidade da adição de graus de liberdade em um contexto local, associados apenas ao elemento enriquecido. No grupo de Enriquecimento de Nós, multiplica-se a função degrau apenas pelas funções de forma padrão associadas aos nós do elemento enriquecido que são separados pela descontinuidade. Assim, o enriquecimento possui um caráter global, pois os graus de liberdade adicionais (que descrevem os deslocamentos entre as faces da descontinuidade) ficam associados aos nós enriquecidos.

Segundo Jirásek (2001), a formação de uma fissura, e conseqüente incorporação de descontinuidades nos campos cinemáticos, é um processo transitório de formação e coalescência de micro fissuras dentro da estrutura material. No estudo da Mecânica da Fratura, divide-se este processo em três estágios, apresentados na Figura 5.2 segundo a descrição do comportamento cinemático da região da fissuração.



**Figura 5.2:** Descrição cinemática da zona de fissuração (Jirásek, 2001).

Na Figura 5.2a, tem-se o estágio denominado *Falha Difusa*. Neste estágio, têm-se os campos de deslocamentos e de deformações ainda contínuos. A localização de deformações é manifestada através de uma alta concentração de deformações em uma banda estreita, com uma transição contínua a partir das deformações mais baixas na região adjacente à banda. Segundo Wolff (2010), em termos físicos, esta representação corresponde a uma zona de processo de fratura com uma alta concentração de defeitos próximos ao seu centro.

Na Figura 5.2b, tem-se o estágio denominado *Descontinuidade Fraca*. Neste estágio, tem-se a evolução do processo à formação de uma região de localização de deformações representada por uma banda de tamanho pequeno, mas finita, separada do restante do corpo por duas *descontinuidades fracas*, ou seja, por saltos no campo de deformações. Observa-se que, apesar da mudança brusca de inclinação, o campo de deslocamentos permanece contínuo. Segundo Wolff (2010), em termos físicos, a banda entre as *descontinuidades fracas* corresponde a uma zona de processo de fratura com uma densidade de microfissuras quase constante.

Na Figura 5.2c, tem-se o estágio denominado *Descontinuidade Forte*. Neste estágio, tem-se o colapso da banda de localização de deformações, incorporando-se uma *descontinuidade forte*, ou seja, um salto no campo de deslocamentos. Consequentemente, o campo de deformações é formado por uma parte regular, obtida pela diferenciação padrão do campo de deslocamentos, e outra parte singular, onde o campo de deformações torna-se infinito. Segundo Wolff (2010), em termos físicos, *descontinuidade forte* corresponde a uma macrofissura com as faces muito fechadas.

A partir da compreensão destes estágios, observa-se que as técnicas descritas para os grupos de Enriquecimento de Elementos e de Nós referem-se à incorporação de *descontinuidades fortes*, tendo em vista que a função degrau é associada às funções de forma, que são empregadas na interpolação do campo de deslocamentos. Conforme mencionado, a fissura pode também ser representada por meio de saltos no campo de deformações, bastando, para tanto, que se incorporem as *descontinuidades fracas*.

Na Figura 5.1, destacam-se as opções de incorporação de *descontinuidades fortes* e *fracas* para o grupo de Enriquecimento de Elementos, ao passo que, para o grupo de Enriquecimento de Nós, destacam-se métodos através dos quais se incorporam descontinuidades nos campos cinemáticos.

Neste trabalho, empregam-se Métodos Constitutivos para a simulação dos estágios de degradação inicial e de localização de deformações e o MEFMG, um Método Cinemático, para a simulação do estágio de incorporação de *descontinuidades fortes*.



### 5.2.1 Métodos Constitutivos

Os Métodos Constitutivos consideram que o meio, apesar de degradado, permanece contínuo, mas que as propriedades do material em estudo se modificam nas regiões submetidas a esforços superiores a determinados parâmetros de resistência.

Vários são os modelos constitutivos propostos segundo esta abordagem, aplicáveis a materiais parcialmente frágeis. Neste trabalho, destacam-se os modelos de fissuras distribuídas e os modelos de dano, nos quais o material é idealizado como um meio contínuo, com degradação devida ao processo de fissuração.

Os modelos de fissuras distribuídas consideram que a região fissurada é formada por um conjunto de pequenas fissuras paralelas entre si. Para tanto, representam esta região através de um conjunto de elementos finitos com comportamento ortotrópico. Conforme Pitangueira (1998), um elemento finito ortotrópico é definido posicionando-se o sistema local do seu tensor constitutivo no sistema definido pelas direções perpendicular e paralela ao plano de fissuração e adotando-se valores diferentes para os módulos de elasticidade destas direções.

Desta forma, a partir de um tensor constitutivo ortotrópico, a rigidez dos elementos da região fissurada representa uma degradação gradual, permitindo-se que os efeitos da fissuração sejam reproduzidos numericamente sem a necessidade de modificações na geometria do modelo e em sua discretização (Wolff, 2010). Conforme Pitangueira (1998), esta degradação gradual é monitorada pelas relações entre tensões e deformações, sendo os limites de resistência do material e parâmetros da Mecânica da Fratura usados na formulação.

Segundo Rots (1988), os modelos de fissuras distribuídas podem ser classificados em duas categorias: modelo com direção de fissuração fixa e modelo com direção de fissuração rotacional. O modelo com direção fixa considera que, uma vez iniciada, a fissura tem orientação fixa durante todo o processo de propagação, ao passo que o modelo com direção rotacional admite que as fissuras podem rotacionar de acordo com a orientação das deformações principais.

Conforme Pituba (1998), os modelos de dano admitem que a perda progressiva de rigidez do material é devida exclusivamente ao processo de microfissuração. Segundo Lemaitre (1992), dano pode ser definido como o processo de surgimento ou nucleação e de avanço de microfissuras e microvazios até a falha do material.

Diversos modelos de dano foram formulados nas últimas décadas. Conforme Pituba (1998), estes modelos são classificados como escalares ou anisotrópicos em função da variável representativa de dano ser, respectivamente, de natureza escalar ou tensorial. Segundo Murakami (2012), o estado de dano pode ser representado por meio de uma variável de dano escalar quando considerado como sendo isotrópico, ou seja, no caso da distribuição aleatória dos microvazios ou da distribuição isotrópica dos vazios esféricos. Por outro lado, quando os microvazios possuem geometria ou configuração orientada, o estado de dano é considerado como sendo anisotrópico e a variável de dano escalar não pode ser aplicada com precisão, empregando-se variáveis de dano tensoriais, a menos que a densidade de vazios seja pequena (Lemaitre, 1992).

Nos modelos de dano, definem-se as variáveis de dano e suas dependências com os tensores constitutivo e de flexibilidade secantes no sentido de representar a dani-ficação. Conforme Penna (2011), os primeiros modelos de dano contínuo tratavam da redução da área de uma seção transversal, a partir de modelos uniaxiais, devido ao surgimento de microfissuras, sendo estas tomadas como a variável de estado interna para a caracterização do dano. Mais recentemente, muitos outros trabalhos desenvolveram modelos de materiais para representar a degradação das propriedades elásticas, ou da rigidez inicial do material, em regime de pequenas deformações. Estes modelos procuram representar a degradação progressiva do módulo de elasticidade quando o material é submetido a estados de tensões e/ou de deformações acima de seus limites de resistência. Segundo Carol et al. (2001), conceitos como tensão e deformação efetivas, bem como equivalência de deformações, de tensões e de energia, são usados para estabelecer os sentidos físicos dos modelos de degradação do material que são as bases das relações secantes com as variáveis de dano.

Dentre as diversas propostas para a evolução do dano, destacam-se as funções escalares de evolução e as superfícies limites de dano. As funções de evolução são definidas em termos de variáveis capazes de medir o estado de tensão e/ou de deformação do material, computando-se assim o estado de degradação correspondente. Dentre estas variáveis, destaca-se a medida de deformação equivalente definida segundo o modelo de dano. Em alternativa, admitindo-se que o domínio elástico do material pode ser delimitado por uma superfície, escrita em termos de parâmetros físicos do meio e de medidas de tensão e/ou de deformação, o valor do dano pode ser obtido usando-se algoritmos de retorno, análogo aos modelos de plasticidade, contudo, em termos de variáveis de dano.

Visando resumir uma grande quantidade de trabalhos devotados à modelagem da degradação dos meios materiais, Carol et al. (1994) propuseram uma teoria unificada de degradação elástica e dano com base em superfície de carregamento. Nesta teoria, a formulação foi desenvolvida em torno da hipótese de uma única função de carregamento. Os modelos de dano escalares mais importantes encontrados na literatura da época foram reinterpretados e reformulados segundo este arcabouço.

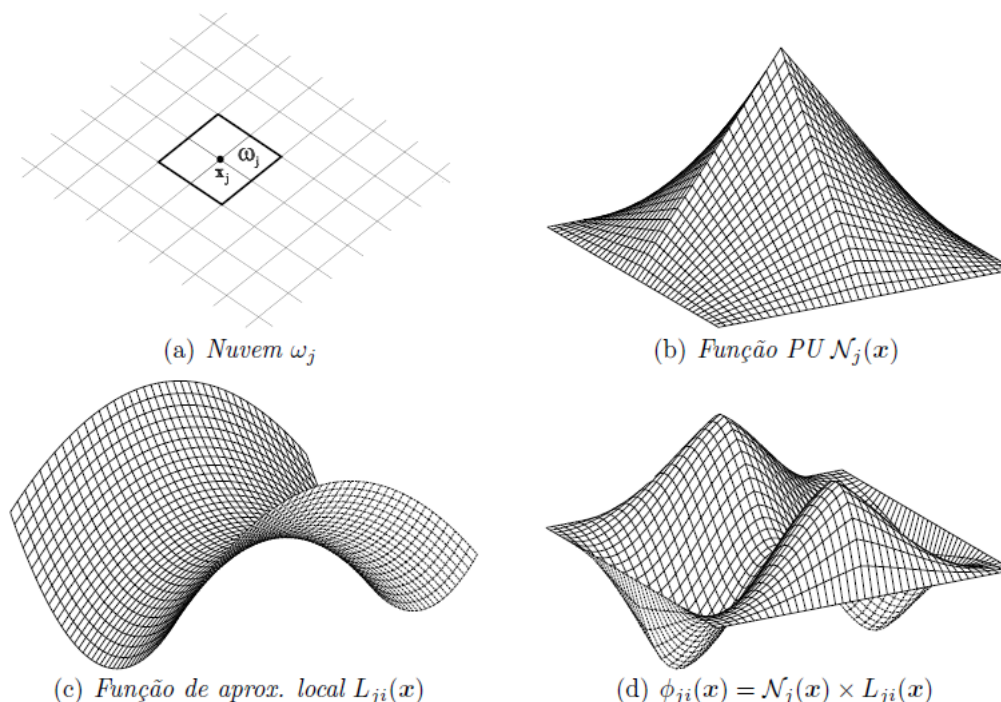
Conforme destacado no apêndice C, um arcabouço teórico e computacional para modelos constitutivos baseados em degradação elástica foi desenvolvido por Penna (2011) no sistema INSANE. Vários modelos clássicos para tratar a degradação do meio material foram abordados no contexto desta estrutura e implementados neste arcabouço. Dentre estes modelos, destacam-se diversos modelos de dano, bem como os modelos de fissuras distribuídas, reformulados considerando-se múltiplas funções de carregamento desacopladas e uma regra de degradação generalizada. Na seção C.3, listam-se alguns dos modelos constitutivos disponibilizados nesta estrutura para a modelagem da degradação do meio de forma distribuída.

Sendo a degradação distribuída do meio o estágio inicial da análise proposta neste trabalho, aproveita-se deste arcabouço para a simulação numérica desta etapa, permitindo-se o emprego de variados modelos constitutivos na modelagem.

### 5.2.2 MEFG - Um Método Cinemático

Proposto de forma independente por Babuska et al. (1994) e Duarte e Oden (1995, 1996), o MEFG pode ser entendido como uma variação do MEF. De acordo com Barros (2002), o emprego sob a denominação atual surge pela primeira vez em Melenk (1995). A estratégia utilizada no MEFG consiste em empregar as funções do tipo Partição da Unidade (PU) que enriquecidas definem as funções de forma. Conforme Barros (2002) e Alves (2012), a escolha das funções de PU depende do tipo de problema a ser analisado. Segundo Alves (2012), o emprego das funções convencionais de MEF como funções de PU, por exemplo, as funções lagrangianas, além de facilitar a aplicação do método, garante estabilidade ao problema analisado, verificando diretamente as condições de contorno.

Para ilustrar a estratégia deste método, considera-se uma malha convencional de elementos finitos bidimensionais definida a partir de um conjunto de  $n$  pontos nodais  $\{\mathbf{x}_j\}_{j=1}^n$ , conforme apresentado na Figura 5.3a. Define-se então a região ou nuvem  $\varpi_j$  formada por todos os elementos que concorrem no ponto nodal  $\mathbf{x}_j$ .



**Figura 5.3:** Estratégia de enriquecimento da nuvem  $\varpi_j$  (Barros, 2002).

O conjunto das funções interpoladoras de Lagrange associadas ao nó  $\mathbf{x}_j$ , obtidas através do MEF, define a função  $N_j(\mathbf{x})$ , cujo suporte corresponde à região  $\varpi_j$ , conforme apresentado na Figura 5.3b.

No MEF, a função  $N_j(\mathbf{x})$ , por si só, define a função de forma, porém, no MEFG, esta função é uma alternativa para compor a função de PU, que enriquecida define a função de forma. Um conjunto  $\mathcal{I}_j$  de funções de enriquecimento, denominadas *funções de aproximação local*, é composto por  $q_j$  funções linearmente independentes definidas para cada nó  $\mathbf{x}_j$ , conforme Equação 5.1. Na Figura 5.3c, ilustra-se uma função de aproximação local  $L_{ji}(\mathbf{x})$ . Conforme Alves (2012), as funções de aproximação local são específicas para um determinado tipo de problema.

$$\mathcal{I}_j \stackrel{def}{=} \{L_{j1}(\mathbf{x}), L_{j2}(\mathbf{x}), \dots, L_{jq}(\mathbf{x})\} = \{L_{ji}(\mathbf{x})\}_{i=1}^q \quad (5.1)$$

com  $L_{j1}(\mathbf{x}) = 1$

Ao final do processo, a função de forma  $\{\phi_{ji}\}_{i=1}^q$  do MEFG, atrelada ao nó  $\mathbf{x}_j$ , é construída por meio do enriquecimento da função de PU pelos componentes do conjunto  $\mathcal{I}_j$ , ou seja,  $\{\phi_{ji}\}_{i=1}^q$  é facilmente obtida através do produto das funções básicas que formam a PU, obtidas através do MEF, pelas funções de enriquecimento, conforme Equação 5.2.

$$\{\phi_{ji}\}_{i=1}^q = N_j(\mathbf{x}) \times \{L_{ji}(\mathbf{x})\}_{i=1}^q \quad (5.2)$$

sem somatório em  $j$ .

Na Figura 5.3d, ilustra-se um componente  $\phi_{ji}(\mathbf{x})$  da função de forma  $\{\phi_{ji}\}_{i=1}^q$ , correspondente à nuvem  $\varpi_j$  apresentada na Figura 5.3a, construído por meio do produto da função de PU, apresentada na Figura 5.3b, pela função de aproximação local  $L_{ji}(\mathbf{x})$ , ilustrada na Figura 5.3c, componente do conjunto de funções de enriquecimento definido na Equação 5.1.

A partir desta estratégia, uma aproximação genérica  $\tilde{\mathbf{u}}(\mathbf{x})$  pode ser obtida por meio da combinação linear das funções de forma, conforme Equação 5.3:

$$\tilde{\mathbf{u}}(\mathbf{x}) = \sum_{j=1}^N N_j(\mathbf{x}) \left\{ \mathbf{u}_j + \sum_{i=2}^q L_{ji}(\mathbf{x}) \mathbf{b}_{ji} \right\} \quad (5.3)$$

onde,

$\mathbf{u}_j$  é o parâmetro nodal convencional, associado à  $N_j(\mathbf{x})$ ;

$\{\mathbf{b}_{ji}\}_{i=2}^q$  são os parâmetros nodais adicionais, associados às  $\{N_j(\mathbf{x}) \cdot L_{ji}(\mathbf{x})\}_{i=2}^q$ .

Conforme Alves (2012), como resultado final do processo, obtém-se a função produto que apresenta as características aproximadoras da *função de aproximação local*, ao mesmo tempo que herda o suporte compacto da PU. Segundo Barros (2002), a utilização deste método proporciona o emprego das funções de MEF para a PU, bem como potencializa a análise de problemas em que o enriquecimento com funções especiais apresenta significativa vantagem.

Conforme destacado no apêndice C, um arcabouço computacional para análise linear pelo MEF foi desenvolvido por Alves (2012) no sistema INSANE. Neste desenvolvimento, incluíram-se as funções de enriquecimento polinomial e de singularidade, bem como a estratégia de enriquecimento global-local. Em Alves et al. (2013), sintetizam-se os detalhes desta implementação, realizada segundo o paradigma de Programação Orientada a Objetos. Em Milione (2014), incluíram-se neste arcabouço as funções de enriquecimento trigonométrico, ao passo que em Wolenski et al. (2014), o expandiram para análise fisicamente não linear.

Devido à generalidade em que foi desenvolvido, em especial para inclusão de novos tipos de função de enriquecimento, este arcabouço pôde ser adaptado para a simulação do estágio de incorporação de *descontinuidades fortes* ao processo, tal como proposto neste trabalho, tendo em vista que esta incorporação consiste em introduzir saltos no campo de deslocamentos, o que pode ser realizado construindo as funções de forma a partir de funções de enriquecimento do tipo degrau.

Neste trabalho, emprega-se a função degrau clássica de *Heaviside*  $H(\xi)$ , definida conforme Equação 5.4. Segundo Mohammadi (2008), diferentes definições foram adotadas para a função de *Heaviside* ao longo dos anos.

$$H(\xi) = \begin{cases} 1, & \forall \xi > 0 \\ 0, & \forall \xi < 0 \end{cases} \quad (5.4)$$

onde,

$\xi$  é a posição em relação à descontinuidade assumida em  $\xi = 0$ .

Desta forma, adotando-se a função de *Heaviside* como função de enriquecimento do MEEG, é possível descrever o campo de deslocamentos decomposto em duas partes, uma contínua e uma descontínua, permitindo-se simular a presença de descontinuidades no modelo. Segundo Mohammadi (2008), a partir desta função de enriquecimento, a Equação 5.3 pode ser reescrita conforme Equação 5.5:

$$\tilde{\mathbf{u}}(x) = \sum_{j=1}^N N_j(\mathbf{x})\mathbf{u}_j + \sum_{k=1}^m N_k(\mathbf{x})H(\xi)\mathbf{b}_k \quad (5.5)$$

onde,

$\mathbf{u}_j$  é o parâmetro nodal convencional, associado à  $N_j(\mathbf{x})$ ;

$\mathbf{b}_k$  é o parâmetro nodal adicional, associado à  $N_k(\mathbf{x})H(\xi)$ ;

$H(\xi)$  é a função degrau clássica de *Heaviside*, também referida como descontínua;

$k$  é o índice dos nós enriquecidos com função degrau.

Segundo Wolff (2010), de fato, a parcela  $\sum_{j=1}^N N_j(\mathbf{x})\mathbf{u}_j$  representa a parte contínua do campo de deslocamentos, enquanto a parcela  $\sum_{k=1}^m N_k(\mathbf{x})\mathbf{b}_k$  representa o salto neste campo ao longo da descontinuidade. Isto pode ser facilmente avaliado, confrontando-se os deslocamentos imediatamente antes e depois da descontinuidade, partindo-se da definição da função de *Heaviside* apresentada na Equação 5.4.

Logo, através desta lógica, têm-se embutidos os efeitos das descontinuidades nas interpolações enriquecidas dos elementos finitos que as contêm.

## Capítulo 6

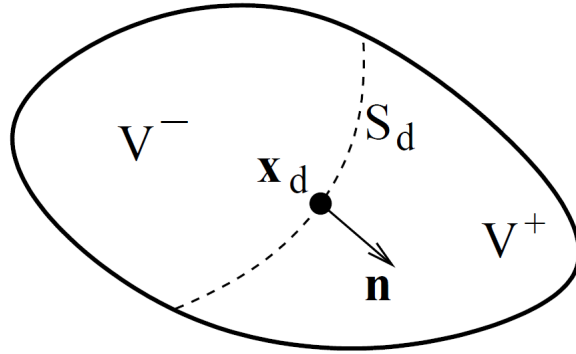
# NUCLEAÇÃO DE FISSURAS - TENSOR ACÚSTICO

Conforme discutido no capítulo 5, a formação de uma fissura é um processo transitório que pode, para fins de estudo, ser dividido em três estágios: *Falha Difusa*, *Descontinuidade Fraca* e *Descontinuidade Forte*.

No estágio de *Descontinuidade Fraca*, tem-se a formação de uma região de localização de deformações delimitada por duas *descontinuidades fracas*. Este estágio antecede o estágio de *Descontinuidade Forte*, no qual a região de localização de deformações colapsa, formando-se uma trinca discreta. Desta forma, o fenômeno de localização de deformações pode ser investigado para caracterizar o estágio limiar de nucleação de fissuras. Segundo Klein e Gao (1998), de fato, este fenômeno pode ser avaliado para indicar o surgimento de trincas. Sendo assim, define-se uma medida capaz de identificar este fenômeno, denominada medida de nucleação de fissuras. Uma importante questão quanto a esta medida é a possibilidade de empregá-la independentemente do modelo constitutivo adotado na análise.

O processo de determinação de uma medida de nucleação apresentado em seguida foi reproduzido de Jirásek (2007a). Neste processo, a análise é restrita a um ponto  $\mathbf{x}_d$  da superfície  $S_d$  da descontinuidade no início da perda de continuidade do campo de deformações. Esta superfície divide o corpo nos subdomínios  $V^+$  e  $V^-$  e tem a sua direção caracterizada por um vetor normal unitário  $\mathbf{n}$  orientado para  $V^+$ , conforme ilustrado na Figura 6.1.





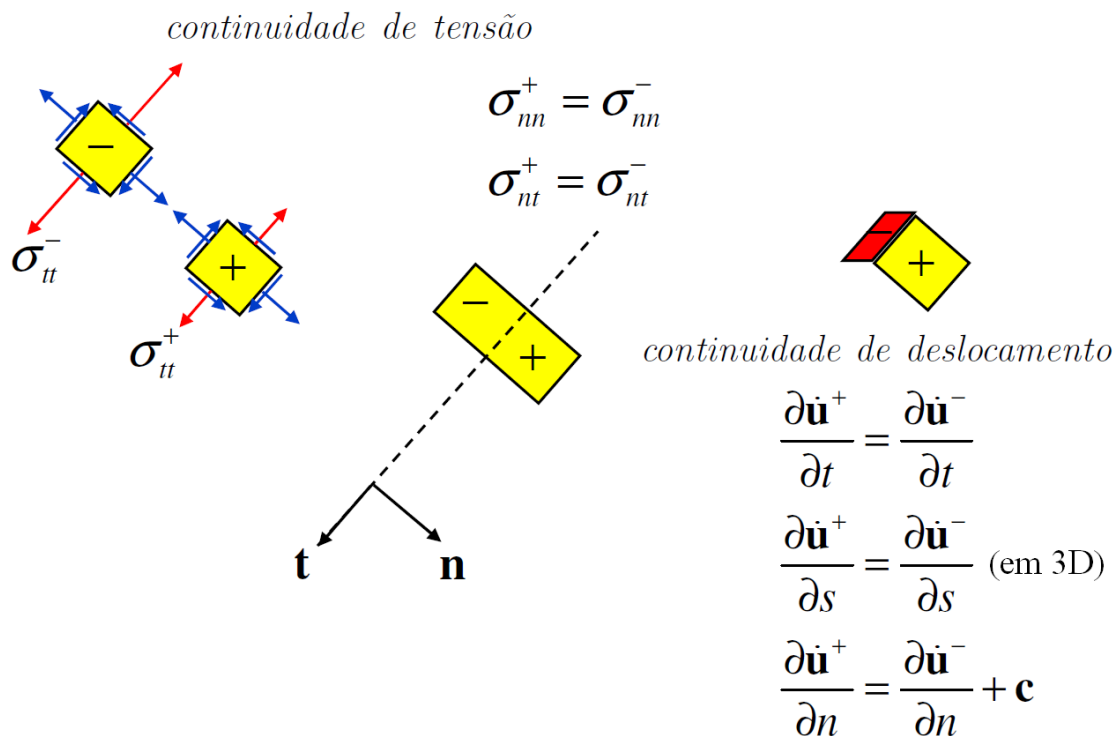
**Figura 6.1:** Corpo cortado por uma descontinuidade (Jirásek, 2007a).

Se o ponto  $\mathbf{x}_d$  é abordado segundo os subdomínios  $V^+$  e  $V^-$ , aqueles campos que são descontínuos através de  $S_d$  tendem a valores diferentes em cada um destes lados, denotados na sequência pelos sobrescritos  $^+$  e  $^-$ , respectivamente. Assim, por exemplo,  $\dot{\boldsymbol{\sigma}}^+$  e  $\dot{\boldsymbol{\sigma}}^-$  denotam, respectivamente, as taxas de tensão nos lados “positivo” e “negativo” da superfície da descontinuidade próximo ao ponto  $\mathbf{x}_d$ , ao passo que a diferença  $[[\dot{\boldsymbol{\sigma}}]] = \dot{\boldsymbol{\sigma}}^+ - \dot{\boldsymbol{\sigma}}^-$  denota o salto na taxa de tensão.

Segundo Jirásek (2007a), ainda que as taxas de tensão e de deformação sejam descontínuas através de  $S_d$ , os correspondentes saltos não são completamente arbitrários. O salto na taxa de tensão  $[[\dot{\boldsymbol{\sigma}}]]$  é restrito pela condição de *continuidade de tensão*, ao passo que o salto na taxa de deformação  $[[\dot{\boldsymbol{\epsilon}}]]$  é restrito pela condição de *continuidade de deslocamento*. Sendo estes tensores decompostos em seus componentes no plano e fora do plano (com respeito ao plano tangente à superfície da descontinuidade), a condição de *continuidade de tensão* significa que as taxas de tensão fora do plano devem ser contínuas, ao passo que a condição de *continuidade de deslocamento* implica que as taxas de deformação no plano devem ser contínuas. Desta forma, os saltos podem aparecer somente nas taxas de tensão no plano e nas taxas de deformação fora do plano.

Segundo Jirásek (2001), as taxas de deformação no plano tangente à superfície da descontinuidade devem ser contínuas, pois, no estágio de *Descontinuidade Fraca*, o campo de deslocamentos permanece contínuo, conforme discutido na seção 5.2.

Na Figura 6.2, ilustram-se a condição de *continuidade de tensão* e a condição de *continuidade de deslocamento*.



**Figura 6.2:** Condições de continuidade de tensão e de deslocamento (Jirásek, 2007b).

A partir da Figura 6.2, a condição de *continuidade de tensão* é facilmente escrita, conforme Equação 6.1:

$$\mathbf{n} \cdot \dot{\boldsymbol{\sigma}}^+ = \mathbf{n} \cdot \dot{\boldsymbol{\sigma}}^- \quad (6.1)$$

Conforme Jirásek (2007a), a representação matemática formal da condição de *continuidade de deslocamento* em termos das taxas de deformação é menos óbvia. Inicia-se das observações que o gradiente espacial de um dado componente do deslocamento, por exemplo, de  $u_1$ , deve ter projeções contínuas sobre o plano tangente à superfície da descontinuidade e que apenas o componente normal deste tensor de primeira ordem pode ter um salto. Ou seja, a partir da Figura 6.2, escreve-se a Equação 6.2:

$$\left(\frac{\partial \dot{u}_1}{\partial \mathbf{x}}\right)^+ = \left(\frac{\partial \dot{u}_1}{\partial \mathbf{x}}\right)^- + c \frac{\partial n}{\partial \mathbf{x}} = \left(\frac{\partial \dot{u}_1}{\partial \mathbf{x}}\right)^- + c \mathbf{n} \quad (6.2)$$

onde,

$c$  é um multiplicador escalar arbitrário.

Estendendo-se a Equação 6.2 a todo o gradiente de deslocamento, tem-se a condição de *continuidade de deslocamento*, conforme Equação 6.3:

$$\left(\frac{\partial \dot{\mathbf{u}}}{\partial \mathbf{x}}\right)^+ = \left(\frac{\partial \dot{\mathbf{u}}}{\partial \mathbf{x}}\right)^- + \mathbf{c} \otimes \frac{\partial n}{\partial \mathbf{x}} = \left(\frac{\partial \dot{\mathbf{u}}}{\partial \mathbf{x}}\right)^- + \mathbf{c} \otimes \mathbf{n} \quad (6.3)$$

onde,

$\mathbf{c}$  é um tensor de primeira ordem arbitrário.

Em teoria de pequenas deformações, com as deformações definidas como a parte simétrica do gradiente de deslocamento, a Equação 6.3 é facilmente reescrita em termos das taxas de deformação, conforme Equação 6.4:

$$\dot{\boldsymbol{\epsilon}}^+ = \dot{\boldsymbol{\epsilon}}^- + (\mathbf{c} \otimes \mathbf{n})_{sim} \quad (6.4)$$

Em seguida, relacionam-se as taxas de tensão e de deformação por meio da lei constitutiva. A lei tensão-deformação na forma de taxa é normalmente escrita como  $\dot{\boldsymbol{\sigma}} = \mathbf{D} : \dot{\boldsymbol{\epsilon}}$ , onde  $\mathbf{D}$  é o tensor constitutivo tangente de quarta ordem, também referido como de rigidez tangente.

A rigidez tangente depende do estado corrente do material, que é inicialmente o mesmo em ambos os lados da descontinuidade incipiente, porém, em alguns casos, pode também depender das taxas de deformação. Uma vez que estas taxas são diferentes nos lados da descontinuidade, admite-se, a priori, que os tensores de rigidez tangente também o são.

Partindo-se deste pressuposto, relacionam-se as taxas de tensão e de deformação nos lados “positivo” e “negativo” da superfície da descontinuidade, respectivamente, conforme Equação 6.5:

$$\dot{\boldsymbol{\sigma}}^+ = \mathbf{D}^+ : \dot{\boldsymbol{\varepsilon}}^+, \quad \dot{\boldsymbol{\sigma}}^- = \mathbf{D}^- : \dot{\boldsymbol{\varepsilon}}^- \quad (6.5)$$

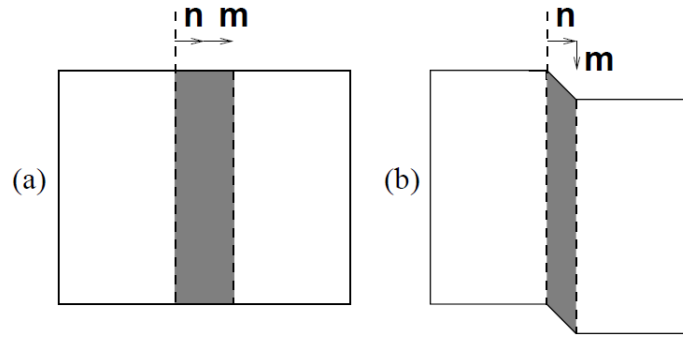
Substituindo-se a Equação 6.5 na condição de *continuidade de tensão*, definida na Equação 6.1, e explorando-se a condição de *continuidade de deslocamento* em termos das taxas de deformação, definida na Equação 6.4, obtém-se a Equação 6.6:

$$\mathbf{n} \cdot \mathbf{D}^+ : \dot{\boldsymbol{\varepsilon}}^- + \overbrace{\mathbf{n} \cdot \mathbf{D}^+ : \mathbf{c}}^{(\mathbf{n} \cdot \mathbf{D}^+ \cdot \mathbf{n}) \cdot \mathbf{c}} = \mathbf{n} \cdot \mathbf{D}^- : \dot{\boldsymbol{\varepsilon}}^- \quad (6.6)$$

Em seguida, move-se para o lado direito da Equação 6.6 o primeiro termo do seu lado esquerdo, obtendo-se a Equação 6.7:

$$(\mathbf{n} \cdot \mathbf{D}^+ \mathbf{n}) \cdot \mathbf{c} = \mathbf{n} \cdot (\mathbf{D}^- - \mathbf{D}^+) : \dot{\boldsymbol{\varepsilon}}^- \quad (6.7)$$

Para atribuir ao tensor  $\mathbf{c}$  um significado físico mais específico, Jirásek (2007a) o apresenta sob a forma  $\mathbf{c} = \dot{\varepsilon} \mathbf{m}$ , onde  $\dot{\varepsilon} = \|\mathbf{c}\|$  é a magnitude do termo salto na Equação 6.3 e  $\mathbf{m} = \mathbf{c} / \|\mathbf{c}\|$  é um tensor de primeira ordem unitário denominado *vetor de polarização*. O ângulo entre os vetores unitários  $\mathbf{m}$  e  $\mathbf{n}$  caracteriza o modo de falha, que vai desde a divisão por tração (modo I), com  $\mathbf{m} = \mathbf{n}$ , até o deslizamento por cisalhamento (modo II), com  $\mathbf{m}$  perpendicular ao  $\mathbf{n}$ , conforme ilustrado nas Figuras 6.3a e 6.3b, respectivamente.



**Figura 6.3:** Modos de falha no plano (a) I e (b) II (Adaptado de Jirásek (2007a)).

A partir desta definição, a Equação 6.7 pode ser reescrita fazendo-se  $\mathbf{c} = \dot{\epsilon}\mathbf{m}$ , conforme Equação 6.8:

$$(\mathbf{n} \cdot \mathbf{D}^+ \mathbf{n}) \cdot \mathbf{m} \dot{\epsilon} = \mathbf{n} \cdot (\mathbf{D}^- - \mathbf{D}^+) : \dot{\epsilon}^- \quad (6.8)$$

Segundo Jirásek (2007a), a Equação 6.8 é a expressão mais geral para descrever o surgimento de uma descontinuidade fraca. Por outro lado, simplificação considerável é alcançada se os tensores de rigidez tangente em ambos os lados da descontinuidade são assumidos os mesmos,  $\mathbf{D}^+ = \mathbf{D}^- = \mathbf{D}$ . Neste caso, o lado direito da Equação 6.8 desaparece e, uma vez que uma descontinuidade verdadeira é obtida somente se  $\dot{\epsilon} \neq 0$ , esta equação se reduz à Equação 6.9:

$$(\mathbf{n} \cdot \mathbf{D} \cdot \mathbf{n}) \cdot \mathbf{m} = 0 \quad (6.9)$$

O tensor de segunda ordem obtido na Equação 6.9 e definido na Equação 6.10 é denominado *tensor de localização*.

$$\mathbf{Q} = \mathbf{n} \cdot \mathbf{D} \cdot \mathbf{n} \quad (6.10)$$

onde,

$\mathbf{n}$  é o vetor normal unitário da superfície da descontinuidade;

$\mathbf{D}$  é o tensor constitutivo tangente, também referido como de rigidez tangente.

Segundo Jirásek (2007a), o *tensor de localização*  $\mathbf{Q}$  é por vezes referido como *tensor acústico*, pois se o tensor de rigidez tangente  $\mathbf{D}$  é tomado como o tensor de rigidez elástico  $\mathbf{D}_e$ , os autovalores do correspondente *tensor acústico*  $\mathbf{Q}_e = \mathbf{n} \cdot \mathbf{D}_e \cdot \mathbf{n}$  dividido pela densidade de massa são os quadrados das velocidades das ondas elásticas propagando na direção de  $\mathbf{n}$ . Os autovetores correspondentes são *vetores de polarização* que determinam os tipos de onda (longitudinal, transversal e mista).

Conforme Jirásek (2007a), a partir da Equação 6.9, conclui-se que, na formação inicial de uma descontinuidade fraca, o *tensor acústico* é singular e o *vetor de polarização*  $\mathbf{m}$  é o autovetor deste tensor associado ao autovalor zero. Esta característica fornece a condição clássica de localização de deformações, definida conforme Equação 6.11.

$$\det \mathbf{Q} = 0 \tag{6.11}$$

Segundo Klein e Gao (1998) e Jirásek (2007a), do ponto de vista matemático, a singularidade do *tensor acústico* indica a chamada *perda de elipticidade da equação diferencial governante*.

O *tensor acústico* definido na Equação 6.10 depende do tensor de rigidez tangente  $\mathbf{D}$  e do vetor normal unitário  $\mathbf{n}$  da superfície da descontinuidade. Sob certas exceções, a rigidez tangente pode ser considerada como dependente somente do estado corrente do material e, portanto, conhecida. O vetor  $\mathbf{n}$ , no entanto, não é conhecido de antemão. Portanto, a análise de localização consiste na busca por um vetor unitário  $\mathbf{n}$  para que o *tensor acústico* se torne singular. Se tal vetor não existir, o campo de deformações deve permanecer contínuo. A singularidade do *tensor acústico* para um determinado vetor  $\mathbf{n}$  indica que um salto no campo de deformações pode se desenvolver através de uma superfície com normal  $\mathbf{n}$ .

Conforme destacado no início deste capítulo, o fenômeno de localização de deformações, caracterizado pela singularidade do *tensor acústico*, é investigado neste trabalho para indicar o estágio limiar de introdução de uma falha discreta.

Ressalta-se que existem várias abordagens para derivar a condição de localização de deformações, tal como apresentada na Equação 6.11. Em Klein e Gao (1998), por exemplo, associam-se a descrição da onda plana e as equações do movimento, tal que a expressão resultante tem a solução não trivial relacionada à singularidade do *tensor acústico*. Optou-se pela abordagem apresentada neste trabalho pelo seu embasamento teórico ser mais didático.

Retornando-se à Equação 6.10 do *tensor acústico*, porém, a partir de seus componentes, tem-se a Equação 6.12:

$$Q_{ik} = D_{ijkl} n_j n_l \quad (6.12)$$

Para o caso de análise plana, o vetor normal unitário  $\mathbf{n}$  da superfície da descontinuidade pode ser definido conforme Equação 6.13:

$$\mathbf{n} = [\cos \psi \quad \text{sen } \psi \quad 0] \quad (6.13)$$

onde,

$\psi$  é o ângulo formado entre as direções  $x$  e do vetor  $\mathbf{n}$ .

Abrindo-se os componentes do *tensor acústico* definidos na Equação 6.12 e empregando-se o vetor  $\mathbf{n}$  definido na Equação 6.13, tem-se a Equação 6.14:

$$\begin{aligned} Q_{11} &= D_{1111} \cos^2 \psi + (D_{1112} + D_{1211}) \cos \psi \text{ sen } \psi + D_{1212} \text{ sen}^2 \psi \\ Q_{12} &= D_{1121} \cos^2 \psi + (D_{1122} + D_{1221}) \cos \psi \text{ sen } \psi + D_{1222} \text{ sen}^2 \psi \\ Q_{13} &= D_{1131} \cos^2 \psi + (D_{1132} + D_{1231}) \cos \psi \text{ sen } \psi + D_{1232} \text{ sen}^2 \psi \\ Q_{21} &= D_{2111} \cos^2 \psi + (D_{2112} + D_{2211}) \cos \psi \text{ sen } \psi + D_{2212} \text{ sen}^2 \psi \\ Q_{22} &= D_{2121} \cos^2 \psi + (D_{2122} + D_{2221}) \cos \psi \text{ sen } \psi + D_{2222} \text{ sen}^2 \psi \\ Q_{23} &= D_{2131} \cos^2 \psi + (D_{2132} + D_{2231}) \cos \psi \text{ sen } \psi + D_{2232} \text{ sen}^2 \psi \\ Q_{31} &= D_{3111} \cos^2 \psi + (D_{3112} + D_{3211}) \cos \psi \text{ sen } \psi + D_{3212} \text{ sen}^2 \psi \\ Q_{32} &= D_{3121} \cos^2 \psi + (D_{3122} + D_{3221}) \cos \psi \text{ sen } \psi + D_{3222} \text{ sen}^2 \psi \\ Q_{33} &= D_{3131} \cos^2 \psi + (D_{3132} + D_{3231}) \cos \psi \text{ sen } \psi + D_{3232} \text{ sen}^2 \psi \end{aligned} \quad (6.14)$$

Conforme discutido em maiores detalhes no capítulo 8, a investigação do fenômeno de localização de deformações para introdução de uma trinca é realizada na convergência de um passo da análise em todos os pontos de Gauss de todos os elementos da malha não cortados por uma descontinuidade já existente. Além disto, conforme justificado também no capítulo 8, a singularidade do *tensor acústico* é usada neste trabalho não somente como a medida para detectar o início da localização de deformações (nucleação de fissuras), mas também como a medida de propagação das trincas, avaliada em todos os pontos de Gauss dos elementos que contêm uma ponta de trinca.

Devido a esta alta demanda e visando um processo menos custoso computacionalmente, o método empregado neste trabalho para investigação da singularidade do *tensor acústico* assume que o tensor de rigidez tangente, definido como  $\mathbf{D} = \mathbf{D}^S + (\partial\mathbf{D}^S/\partial\boldsymbol{\varepsilon}) \boldsymbol{\varepsilon}$ , tem como componentes não nulos somente aqueles formados por um componente não nulo do tensor de rigidez secante, por sua vez definido assumindo-se a ortotropia do meio. Ou seja, assume-se que as posições não nulas do tensor de rigidez tangente  $\mathbf{D}$  são as mesmas não nulas do tensor de rigidez secante  $\mathbf{D}^S$ , desprezando-se a possível contribuição da parcela  $(\partial\mathbf{D}^S/\partial\boldsymbol{\varepsilon}) \boldsymbol{\varepsilon}$  para os demais componentes. Partindo-se deste pressuposto, o tensor de rigidez tangente é assumido neste método conforme Equação 6.15:

$$\mathbf{D} = \begin{bmatrix} D_{1111} & 0 & 0 & 0 & D_{1212} & 0 & 0 & 0 & D_{1313} \\ 0 & D_{1122} & 0 & D_{1221} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & D_{1133} & 0 & 0 & 0 & D_{1331} & 0 & 0 \\ \hline 0 & D_{2112} & 0 & D_{2211} & 0 & 0 & 0 & 0 & 0 \\ D_{2121} & 0 & 0 & 0 & D_{2222} & 0 & 0 & 0 & D_{2323} \\ 0 & 0 & 0 & 0 & 0 & D_{2233} & 0 & D_{2332} & 0 \\ \hline 0 & 0 & D_{3113} & 0 & 0 & 0 & D_{3311} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & D_{3223} & 0 & D_{3322} & 0 \\ D_{3131} & 0 & 0 & 0 & D_{3232} & 0 & 0 & 0 & D_{3333} \end{bmatrix} \quad (6.15)$$



A partir da Equação 6.14 e da suposição acerca do tensor de rigidez tangente, assumido conforme Equação 6.15, reescrevem-se os componentes não nulos do *tensor acústico* para o caso de análise plana, conforme Equação 6.16, e a condição clássica de localização de deformações, definida na Equação 6.11, conforme Equação 6.17.

$$\begin{aligned}
Q_{11} &= D_{1111} \cos^2 \psi + D_{1212} \sin^2 \psi \\
Q_{12} &= (D_{1122} + D_{1221}) \cos \psi \sin \psi \\
Q_{21} &= (D_{2112} + D_{2211}) \cos \psi \sin \psi \\
Q_{22} &= D_{2121} \cos^2 \psi + D_{2222} \sin^2 \psi \\
Q_{33} &= D_{3131} \cos^2 \psi + D_{3232} \sin^2 \psi
\end{aligned} \tag{6.16}$$

$$\det \mathbf{Q} = 0 \quad \therefore (Q_{11}Q_{22} - Q_{21}Q_{12})Q_{33} = 0 \quad \therefore Q_{11}Q_{22} - Q_{21}Q_{12} = 0 \tag{6.17}$$

Os componentes do *tensor acústico* necessários à Equação 6.17 podem, ainda, ser reescritos conforme Equação 6.18:

$$\begin{aligned}
Q_{11} &= a_1 \cos^2 \psi + b_1 \sin^2 \psi = (a_1 - b_1) \cos^2 \psi + b_1 \\
Q_{22} &= c_1 \cos^2 \psi + d_1 \sin^2 \psi = (c_1 - d_1) \cos^2 \psi + d_1 \\
Q_{12} &= e_1 \cos \psi \sin \psi \\
Q_{21} &= f_1 \cos \psi \sin \psi
\end{aligned} \tag{6.18}$$

onde,

$$a_1 = D_{1111}$$

$$b_1 = D_{1212}$$

$$c_1 = D_{2121}$$

$$d_1 = D_{2222}$$

$$e_1 = D_{1122} + D_{1221}$$

$$f_1 = D_{2112} + D_{2211}$$

Substituindo-se os componentes do *tensor acústico*, definidos conforme Equação 6.18, na condição de localização de deformações, definida conforme Equação 6.17, obtém-se a Equação 6.19:

$$a \cos^4 \psi + b \cos^2 \psi + c = 0 \quad \therefore \quad ax^2 + bx + c = 0 \quad \text{com} \quad x = \cos^2 \psi \quad (6.19)$$

onde,

$$a = (a_1 - b_1)(c_1 - d_1) + e_1 f_1$$

$$b = a_1 d_1 + b_1 c_1 - 2b_1 d_1 - e_1 f_1$$

$$c = b_1 d_1$$

Portanto, através deste método, a análise de localização consiste na busca pelas raízes do polinômio do segundo grau definido na Equação 6.19, facilmente realizada através da fórmula de Bhaskara. Se este polinômio tiver raiz real e a raiz quadrada desta raiz, que define o  $\cos \psi$ , estiver compreendida entre 0 e 1, tem-se a identificação do fenômeno de localização de deformações. Esta formulação baseia-se naquela desenvolvida por Klein e Gao (1998) e empregada em Thiagarajan et al. (2004).

Ressalta-se que a determinação do  $\cos \psi$ , por si só, não define a direção do vetor normal unitário  $\mathbf{n}$ , mas sim o sistema definido por este vetor e a direção da trinca. Desta forma, neste trabalho, a singularidade do *tensor acústico* é empregada para identificar o fenômeno de localização de deformações, mas não para definir a direção da trinca. Conforme detalhado no capítulo 8, a direção da trinca é assumida como sendo a direção normal à máxima deformação principal não local.

No sistema INSANE, este método foi implementado sob o nome *checksStrain-Localization* na classe modelo de análise *Plane*, por ser especializado para análise plana. Tal método recebe como argumento o tensor de rigidez tangente.

Destaca-se que a praticidade deste método, que otimiza a análise proposta neste trabalho, é devida à simplicidade da expressão definida na Equação 6.19, por sua vez alcançada pela consideração acerca do tensor de rigidez tangente.

Por outro lado, poderia-se, ao invés de prever quais são os termos não nulos do tensor de rigidez tangente, fazer uma busca pela singularidade do *tensor acústico* variando-se o vetor  $\mathbf{n}$ , porém, seria mais custoso computacionalmente. Esta proposta poderia ainda ser otimizada iniciando-se a busca a partir da direção da máxima deformação principal, assumindo-se que esta direção tem uma variação pequena em relação à do vetor  $\mathbf{n}$  com o qual se pode ter um *tensor acústico* singular.

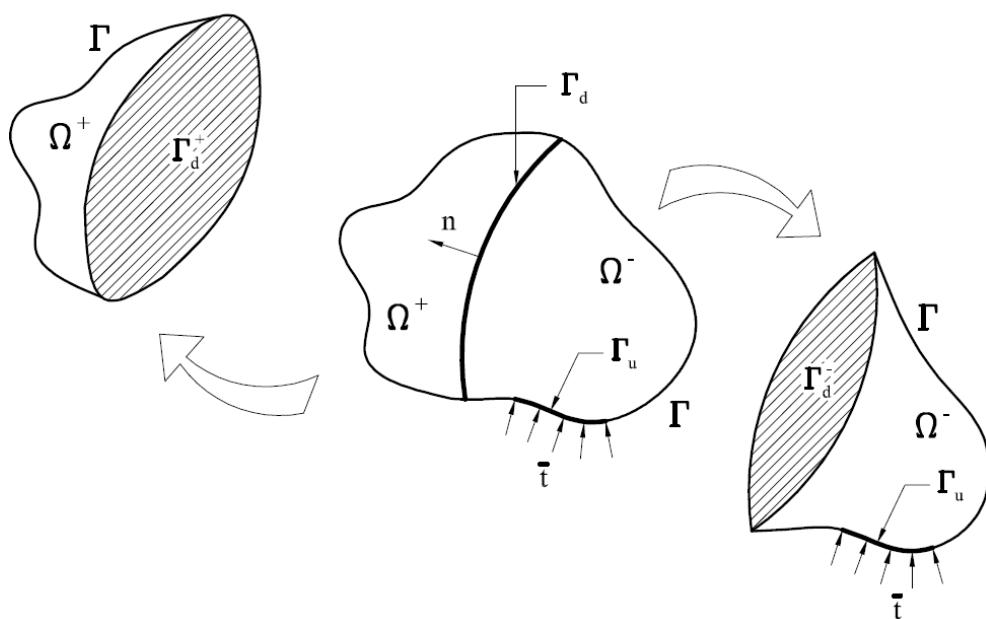
O emprego da singularidade do *tensor acústico* como medida de nucleação de fissuras destaca-se pela sua aplicação independentemente do modelo constitutivo adotado na análise. Porém, no caso do modelo constitutivo linear elástico, é evidente que esta medida não é capaz de identificar o fenômeno de localização de deformações.

Desta forma, para análise de nucleação e propagação de trincas em meio linear elástico, emprega-se a resistência à tração do material como medida de nucleação e de propagação, comparando-a à máxima tensão principal. Para tanto, adota-se o material *ConcreteCohesiveCrack* desenvolvido por Wolff (2010) no sistema INSANE, o qual, dentre os parâmetros de entrada, tem a resistência à tração.

## Capítulo 7

# MODELO CONSTITUTIVO DISCRETO

Neste capítulo, um modelo constitutivo discreto, responsável por simular as forças de coesão atuantes nos planos das trincas, é formulado para uma lei coesiva particular, porém, sem perda da generalidade do sistema. Como modelo constitutivo discreto, entende-se a relação entre tensões e deslocamentos no caminho da trinca. Primeiramente, retorna-se ao corpo cortado por uma descontinuidade apresentado na Figura 6.1, reapresentado conforme Figura 7.1.



**Figura 7.1:** Corpo  $\Omega$  cortado por uma descontinuidade  $\Gamma_d$  (Wells e Sluys, 2001).

Segundo Wells e Sluys (2001), o campo de deslocamentos  $\mathbf{u}$  do corpo  $\Omega$  ( $\Omega^+ \cup \Omega^-$ ) pode ser decomposto em duas partes, uma contínua e a outra descontínua, conforme Equação 7.1:

$$\mathbf{u}(\mathbf{x}, t) = \hat{\mathbf{u}}(\mathbf{x}, t) + H[[\mathbf{u}(\mathbf{x}, t)]] \quad (7.1)$$

onde,

$\hat{\mathbf{u}}$  e  $[[\mathbf{u}]]$  são funções contínuas em  $\Omega$ ;

$H$  é a função de *Heaviside*, apresentada na subseção 5.2.2.

A função de *Heaviside* é centralizada na descontinuidade  $\Gamma_d$  do corpo  $\Omega$ , tal que  $H = 1 \quad \forall \quad \mathbf{x} \in \Omega^+$  e  $H = 0 \quad \forall \quad \mathbf{x} \in \Omega^-$ . Desta forma, os componentes do salto de deslocamentos na descontinuidade são descritos por  $[[\mathbf{u}]]_{\mathbf{x} \in \Gamma_d}$ .

O campo de deformações do corpo  $\Omega$  cortado pela descontinuidade, por sua vez, é obtido calculando-se o gradiente da Equação 7.1, conforme Equação 7.2:

$$\begin{aligned} \boldsymbol{\varepsilon} &= \nabla^s \hat{\mathbf{u}} + H(\nabla^s [[\mathbf{u}]]) + (\nabla H \otimes [[\mathbf{u}]])^s \\ &= \underbrace{\nabla^s \hat{\mathbf{u}} + H(\nabla^s [[\mathbf{u}]])}_{\text{limitado}} + \underbrace{\delta_{\Gamma_d} ([[ \mathbf{u} ]]) \otimes \mathbf{n}}_{\text{não-limitado}}^s \end{aligned} \quad (7.2)$$

onde,

$\delta_{\Gamma_d}$  é a função delta de *Dirac* centralizada na descontinuidade;

$\mathbf{n}$  é o vetor normal unitário da superfície da descontinuidade orientado para  $\Omega^+$ ;

$()^s$  indica que somente a parte simétrica dos tensores está sendo considerada.

Conforme Wolff (2010), o delta de *Dirac* é uma distribuição ou medida matemática que, informalmente, consiste em uma função generalizada  $\delta(x)$  que vale zero para qualquer  $x$ , exceto para  $x = 0$ , onde seu valor tende ao infinito, de tal maneira que o valor total de sua integral vale 1.

A função delta de *Dirac* aparece no campo de deformações devido ao salto imposto no campo de deslocamentos. Uma vez que esta função não é limitada (tem valor infinito), torna-se impossível lhe atribuir um sentido físico, entretanto, esta função se torna bastante útil na inserção do campo de deformações na equação dos trabalhos virtuais.

A partir das Equações 7.1 e 7.2, os campos de deslocamentos e de deformações de um elemento finito cortado por uma descontinuidade podem ser interpolados, respectivamente, conforme Equações 7.3 e 7.4:

$$\mathbf{u}(\mathbf{x}) = \underbrace{\mathbf{N}(\mathbf{x})\mathbf{a}}_{\hat{\mathbf{u}}} + H \underbrace{\mathbf{N}(\mathbf{x})\mathbf{b}}_{[\mathbf{u}]} \quad (7.3)$$

$$\boldsymbol{\varepsilon} = \mathbf{B}\mathbf{a} + H\mathbf{B}\mathbf{b} + (\delta_{\Gamma_d}\mathbf{n})\mathbf{N}\mathbf{b} \quad (7.4)$$

onde,

$\mathbf{N}$  é a matriz das funções de forma convencionais de MEF;

$\mathbf{a}$  é o vetor dos graus de liberdade nodal convencionais;

$H$  é a função de *Heaviside*;

$\mathbf{b}$  é o vetor dos graus de liberdade nodal adicionais;

$\mathbf{B}$  é matriz de aproximação das deformações;

$\delta_{\Gamma_d}$  é a função delta de *Dirac*;

$\mathbf{n}$  é a matriz dos componentes do vetor normal à superfície da descontinuidade.

Conforme Wells e Sluys (2001), a matriz  $\mathbf{n}$ , para o caso bidimensional, é definida conforme Equação 7.5:

$$\mathbf{n} = \begin{bmatrix} n_x & 0 \\ 0 & n_y \\ n_y & n_x \end{bmatrix} \quad (7.5)$$

Ressalta-se que a Equação 7.3 é a Equação 5.5 na forma matricial e sob notações diferentes. Conforme destacado na subseção 5.2.2, efetivamente, a parcela da Equação 7.3 associada aos graus de liberdade convencionais representa a parte contínua do campo de deslocamentos, enquanto a parcela associada aos graus de liberdade adicionais representa o salto neste campo ao longo da descontinuidade. A introdução da função de *Heaviside* na equação de interpolação do campo de deslocamentos dos elementos finitos resulta em um salto de deslocamentos ao longo da descontinuidade de mesmo grau polinomial que as funções de forma.

Na sequência, considera-se a equação dos trabalhos virtuais, desprezando-se as forças de corpo, conforme Equação 7.6:

$$\int_{\Omega} \nabla^s \boldsymbol{\eta} : \boldsymbol{\sigma} \, d\Omega = \int_{\Gamma_u} \boldsymbol{\eta} \cdot \bar{\mathbf{t}} \, d\Gamma \quad (7.6)$$

onde,

$\boldsymbol{\eta}$  são as variações admissíveis de deslocamentos;

$\boldsymbol{\sigma}$  é o campo de tensões;

$\bar{\mathbf{t}}$  são as forças relacionadas às tensões externas aplicadas no contorno  $\Gamma_u$ .

Inserindo-se na Equação 7.6 os campos de deslocamentos e de deformações definidos, respectivamente, nas Equações 7.1 e 7.2, obtém-se a Equação 7.7:

$$\begin{aligned} \int_{\Omega} \nabla^s \hat{\boldsymbol{\eta}} : \boldsymbol{\sigma} \, d\Omega + \int_{\Omega} H(\nabla^s [\boldsymbol{\eta}]) : \boldsymbol{\sigma} \, d\Omega + \int_{\Omega} \delta_{\Gamma_d}([\boldsymbol{\eta}] \otimes \mathbf{n})^s : \boldsymbol{\sigma} \, d\Omega = \\ = \int_{\Gamma_u} \hat{\boldsymbol{\eta}} \cdot \bar{\mathbf{t}} \, d\Gamma + \int_{\Gamma_u} H[\boldsymbol{\eta}] \cdot \bar{\mathbf{t}} \, d\Gamma \end{aligned} \quad (7.7)$$

Em seguida, integrando-se a distribuição delta de *Dirac* na Equação 7.7 ao longo do corpo  $\Omega$ , o termo não limitado desaparece (por definição, a integral da função delta de *Dirac* tem valor unitário), obtendo-se a Equação 7.8:

$$\begin{aligned}
\int_{\Omega} \nabla^s \hat{\boldsymbol{\eta}} : \boldsymbol{\sigma} \, d\Omega + \int_{\Omega} H(\nabla^s \llbracket \boldsymbol{\eta} \rrbracket) : \boldsymbol{\sigma} \, d\Omega + \int_{\Gamma_d} \llbracket \boldsymbol{\eta} \rrbracket_{\Gamma_d} \cdot \mathbf{t} \, d\Gamma = \\
= \int_{\Gamma_u} \hat{\boldsymbol{\eta}} \cdot \bar{\mathbf{t}} \, d\Gamma + \int_{\Gamma_u} H \llbracket \boldsymbol{\eta} \rrbracket \cdot \bar{\mathbf{t}} \, d\Gamma
\end{aligned} \tag{7.8}$$

onde,

$\llbracket \boldsymbol{\eta} \rrbracket_{\Gamma_d}$  é a variação do salto de deslocamentos na descontinuidade  $\Gamma_d$ ;

$\mathbf{t} = (\boldsymbol{\sigma} \cdot \mathbf{n})_{\Gamma_d}$  é a tensão coesiva agindo na descontinuidade.

A seguir, para que a matriz de rigidez e os vetores de forças sejam obtidos, as interpolações devem ser inseridas na equação dos trabalhos virtuais. Para tanto, a partir da Equação 7.3, definem-se as variações  $\hat{\boldsymbol{\eta}}$  e  $\llbracket \boldsymbol{\eta} \rrbracket$  em termos das variações dos deslocamentos nodais, respectivamente, conforme Equações 7.9 e 7.10:

$$\hat{\boldsymbol{\eta}} = \mathbf{N} \mathbf{a}' \tag{7.9}$$

onde,

$\mathbf{a}'$  são as variações dos graus de liberdade nodal  $\mathbf{a}$ .

$$\llbracket \boldsymbol{\eta} \rrbracket = \mathbf{N} \mathbf{b}' \tag{7.10}$$

onde,

$\mathbf{b}'$  são as variações dos graus de liberdade nodal  $\mathbf{b}$ .

A partir das Equações 7.9 e 7.10, escrevem-se os gradientes das variações  $\hat{\boldsymbol{\eta}}$  e  $\llbracket \boldsymbol{\eta} \rrbracket$  na forma discreta, respectivamente, conforme Equações 7.11 e 7.12:

$$\nabla^s \hat{\boldsymbol{\eta}} = \mathbf{B} \mathbf{a}' \tag{7.11}$$

$$\nabla^s \llbracket \boldsymbol{\eta} \rrbracket = \mathbf{B} \mathbf{b}' \tag{7.12}$$



Inserindo-se na Equação 7.8 a forma discreta das variações dos deslocamentos definida nas Equações 7.9 e 7.10, bem como a dos seus gradientes definida nas Equações 7.11 e 7.12, e separando-se os termos relativos a  $\hat{\boldsymbol{\eta}}$  e  $[[\boldsymbol{\eta}]]$  na forma de um sistema de equações, obtêm-se, respectivamente, as Equações 7.13 e 7.14:

$$\int_{\Omega} \mathbf{B}^T \boldsymbol{\sigma} \, d\Omega = \int_{\Gamma_u} \mathbf{N}^T \bar{\mathbf{t}} \, d\Gamma \quad (7.13)$$

$$\int_{\Omega} H\mathbf{B}^T \boldsymbol{\sigma} \, d\Omega + \int_{\Gamma_d} \mathbf{N}^T \mathbf{t} \, d\Gamma = \int_{\Gamma_u} H\mathbf{N}^T \bar{\mathbf{t}} \, d\Gamma \quad (7.14)$$

A partir das Equações 7.13 e 7.14, os vetores de forças equivalentes às tensões internas correspondentes às variações dos graus de liberdade convencionais  $\mathbf{a}$  e adicionais  $\mathbf{b}$  são escritos, respectivamente, conforme Equações 7.15 e 7.16:

$$\mathbf{f}_a^{\text{int}} = \int_{\Omega} \mathbf{B}^T \boldsymbol{\sigma} \, d\Omega \quad (7.15)$$

$$\mathbf{f}_b^{\text{int}} = \int_{\Omega} H\mathbf{B}^T \boldsymbol{\sigma} \, d\Omega + \int_{\Gamma_d} \mathbf{N}^T \mathbf{t} \, d\Gamma \quad (7.16)$$

onde,

“a” e “b” denotam as variações das partes convencionais e adicionais, respectivamente.

Conforme Wells e Sluys (2001), para se desenvolver um procedimento incremental da solução de maneira eficiente, as relações constitutivas devem ser colocadas na forma de taxa ou razão. A taxa de tensão no domínio do contínuo pode ser expressa em termos das velocidades dos deslocamentos nodais, conforme Equação 7.17:

$$\dot{\boldsymbol{\sigma}} = \mathbf{D}\dot{\boldsymbol{\epsilon}} = \mathbf{D}(\mathbf{B}\dot{\mathbf{a}} + H\mathbf{B}\dot{\mathbf{b}}) \quad (7.17)$$

onde,

$\mathbf{D}$  é a matriz constitutiva tangente, também referida como de rigidez tangente.

Analogamente à taxa de tensão no domínio do contínuo, a taxa de tensão coesiva na descontinuidade pode ser expressa em termos das velocidades dos deslocamentos nodais adicionais, conforme Equação 7.18:

$$\dot{\mathbf{t}} = \mathbf{T}[[\dot{\mathbf{u}}]]_{\mathbf{x} \in \Gamma_d} = \mathbf{TN}\dot{\mathbf{b}} \quad (7.18)$$

onde,

$\mathbf{T}$  é a matriz de rigidez tangente da relação entre tensão coesiva e deslocamento na descontinuidade.

Ao final deste capítulo, obtém-se, para uma lei coesiva particular, a rigidez tangente  $\mathbf{T}$  a partir da formulação de um modelo constitutivo discreto.

A substituição das Equações 7.17 e 7.18 na equação discreta dos trabalhos virtuais, dada pelas Equações 7.13 e 7.14, conduz à obtenção da matriz de rigidez  $\mathbf{K}$  em função dos deslocamentos incrementais  $\mathbf{da}$  e  $\mathbf{db}$ , conforme Equação 7.19:

$$\underbrace{\begin{bmatrix} \int_{\Omega} \mathbf{B}^T \mathbf{DB} \, d\Omega & \int_{\Omega} H \mathbf{B}^T \mathbf{DB} \, d\Omega \\ \int_{\Omega} H \mathbf{B}^T \mathbf{DB} \, d\Omega & \int_{\Omega} H^2 \mathbf{B}^T \mathbf{DB} \, d\Omega + \int_{\Gamma_d} \mathbf{N}^T \mathbf{TN} \, d\Gamma \end{bmatrix}}_{\mathbf{K}} \begin{Bmatrix} \mathbf{da} \\ \mathbf{db} \end{Bmatrix} = \begin{Bmatrix} \mathbf{f}_a^{\text{ext}} \\ \mathbf{f}_b^{\text{ext}} \end{Bmatrix} - \begin{Bmatrix} \mathbf{f}_a^{\text{int}} \\ \mathbf{f}_b^{\text{int}} \end{Bmatrix} \quad (7.19)$$

Na Equação 7.19,  $\mathbf{f}_a^{\text{ext}}$  e  $\mathbf{f}_b^{\text{ext}}$  são os vetores de forças equivalentes às forças externas correspondentes, respectivamente, às variações dos graus de liberdade convencionais  $\mathbf{a}$  e adicionais  $\mathbf{b}$ . Estes vetores são definidos, respectivamente, no lado direito das Equações 7.13 e 7.14 e rerepresentados nas Equações 7.20 e 7.21:

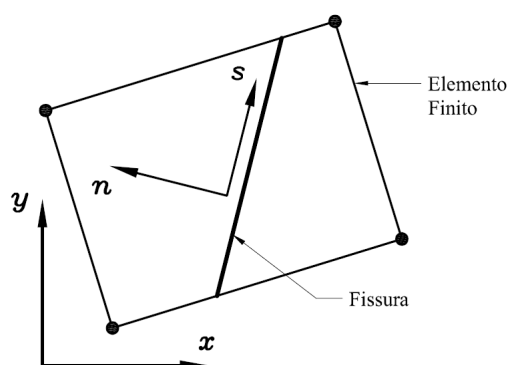
$$\mathbf{f}_a^{\text{ext}} = \int_{\Gamma_u} \mathbf{N}^T \bar{\mathbf{t}} \, d\Gamma \quad (7.20)$$

$$\mathbf{f}_b^{\text{ext}} = \int_{\Gamma_u} H \mathbf{N}^T \bar{\mathbf{t}} \, d\Gamma \quad (7.21)$$

Ressalva-se que, neste trabalho, a configuração particionada da matriz de rigidez  $\mathbf{K}$  e dos vetores de deslocamentos e de forças do elemento em função do grau de liberdade nodal ser convencional ou adicional, tal como apresentada na Equação 7.19, é meramente ilustrativa. Conforme discutido na subseção 5.2.2, o processo de enriquecimento nodal, e conseqüente adição de graus de liberdade ao elemento, é realizado por meio do arcabouço de análise pelo MEEFG desenvolvido por Alves (2012) no sistema INSANE, no qual a montagem da matriz de rigidez do elemento, bem como dos seus vetores de deslocamentos e de forças, é realizada segundo a seqüência de incidência nodal e a ordem dos correspondentes graus de liberdade, abrangendo-se os convencionais e os adicionais.

As rigidezes tangentes  $\mathbf{D}$  e  $\mathbf{T}$ , necessárias à construção da matriz de rigidez  $\mathbf{K}$  do elemento, conforme Equação 7.19, são funções, respectivamente, do modelo constitutivo distribuído, abordado na subseção 5.2.1, e do modelo constitutivo discreto.

O modelo constitutivo discreto empregado neste trabalho baseia-se no conceito de fissura coesiva (ou fictícia), em particular no modelo de Hillerborg et al. (1976), no qual todas as deformações inelásticas, que ocorrem na zona de processo de fratura, são representadas em uma linha através das forças coesivas que agem na trinca, ou em uma extensão fictícia desta. Este modelo é definido por um sistema de coordenadas local, ortogonal, no qual os vetores  $\mathbf{n}$  e  $\mathbf{s}$  representam, respectivamente, as direções normal e paralela à fissura, conforme ilustrado na Figura 7.2.



**Figura 7.2:** Sistema de coordenadas local da descontinuidade (Wolff, 2010).

Para o caso de análise plana, o modelo constitutivo discreto considera que o dano na região da fissura envolve os modos de falha I e II, definidos, respectivamente, como divisão por tração e deslizamento por cisalhamento. Sendo assim, define-se a rigidez tangente  $\mathbf{T}$  relacionando-a à separação normal e ao deslizamento tangencial relativo das superfícies da fissura. Para tanto, assume-se que o componente normal  $t_n$  da tensão coesiva é função da máxima separação normal observada das superfícies da fissura, ao passo que o componente tangencial  $t_s$  é função do deslizamento tangencial relativo destas superfícies. Partindo-se deste pressuposto, inicia-se definindo a função de carregamento  $f$ , dada pela Equação 7.22:

$$f(\llbracket u \rrbracket_n, \kappa) = \llbracket u \rrbracket_n - \kappa \quad (7.22)$$

onde,

$\llbracket u \rrbracket_n$  é a separação normal entre as superfícies da fissura;

$\kappa$  é um parâmetro histórico, dado pelo maior valor de  $\llbracket u \rrbracket_n$  alcançado na análise.

Através da Equação 7.22, a abertura da fissura, e conseqüente carregamento, é indicada por  $f \geq 0$ , ao passo que o seu fechamento, e conseqüente descarregamento, por  $f < 0$ . Destaca-se que uma fissura fechada é indicada por  $\llbracket u \rrbracket_n < 0$ .

Definida a função de carregamento, na seqüência, definem-se os componentes normal  $t_n$  e tangencial  $t_s$  da lei coesiva empregada neste trabalho.

O componente normal  $t_n$  é aproximado por uma função exponencial decrescente do parâmetro histórico  $\kappa$ , conforme Equação 7.23:

$$t_n = f_t \exp\left(-\frac{f_t}{G_f} \kappa\right) \quad (7.23)$$

onde,

$f_t$  é a resistência à tração do material;

$G_f$  é a energia de fratura.

Neste trabalho, emprega-se uma simplificação proposta por Wells e Sluys (2001) acerca do componente tangencial  $t_s$ , com vantagens computacionais significativas. Assume-se, a priori, que este componente é aproximado pela Equação 7.24:

$$t_s = d_{\text{init}} \exp(h_s \kappa) \llbracket u \rrbracket_s \quad (7.24)$$

onde,

$d_{\text{init}}$  é a rigidez inicial da fissura ao cisalhamento, definida para  $\kappa = 0$ ;

$h_s$  é um parâmetro relacional de rigidezes da fissura ao cisalhamento;

$\llbracket u \rrbracket_s$  é o deslizamento tangencial relativo das superfícies da fissura.

O parâmetro relacional de rigidezes da fissura ao cisalhamento  $h_s$  é definido conforme Equação 7.25:

$$h_s = \ln(d_{\kappa=1,0}/d_{\text{init}}) \quad (7.25)$$

onde,

$d_{\kappa=1,0}$  é a rigidez da fissura ao cisalhamento, definida para  $\kappa = 1, 0$ .

Conforme Wells e Sluys (2001), o componente tangencial  $t_s$  em função da máxima separação normal observada das superfícies da fissura produz uma rigidez tangente  $\mathbf{T}$  não simétrica e, conseqüentemente, uma matriz de rigidez  $\mathbf{K}$  do elemento também não simétrica. Este empecilho é contornado, assumindo-se que a rigidez da fissura ao cisalhamento é constante, ou seja, é igual à  $d_{\text{init}}$ . Desta forma, a Equação 7.24 pode ser reescrita conforme Equação 7.26:

$$t_s = d_{\text{init}} \llbracket u \rrbracket_s \quad (7.26)$$

onde,

$d_{\text{init}}$  é a rigidez inicial da fissura ao cisalhamento, assumida constante.

Segundo Wells e Sluys (2001), o risco que se corre ao se assumir a rigidez da fissura ao cisalhamento constante é que, se esta for feita muito pequena, uma resposta global pós-pico excessivamente frágil será prevista, ao passo que, se feita muito grande, um bloqueio de tensão será observado, ou seja, a tensão coesiva oferecerá muita resistência à abertura da fissura, mantendo-a sob um fechamento excessivo.

Neste modelo, o comportamento de amolecimento na interface é dirigido somente pela separação normal das superfícies da fissura, embora saiba-se que, à medida em que estas superfícies se separam, a rigidez da fissura ao cisalhamento tende à zero.

Por fim, tomam-se os diferenciais das Equações 7.23 e 7.26 em relação ao tempo, obtendo-se a rigidez tangente  $\mathbf{T}$ , conforme Equação 7.27:

$$\begin{Bmatrix} \dot{t}_n \\ \dot{t}_s \end{Bmatrix} = \underbrace{\begin{bmatrix} -\frac{f_t^2}{G_f} \exp\left(-\frac{f_t}{G_f} \kappa\right) & 0 \\ 0 & d_{\text{init}} \end{bmatrix}}_{\mathbf{T}} \begin{Bmatrix} \llbracket \dot{u} \rrbracket_n \\ \llbracket \dot{u} \rrbracket_s \end{Bmatrix} \quad (7.27)$$

Ressalta-se que a rigidez tangente  $\mathbf{T}$  é empregada quando a fissura experimenta carregamento, indicado por  $f \geq 0$ . No caso de descarregamento, indicado por  $f < 0$ , o comportamento é simulado por meio da rigidez secante  $\mathbf{S}$ , por sua vez obtida dividindo-se os componentes da tensão coesiva, definidos nas Equações 7.23 e 7.26, pela correspondente variável, conforme Equação 7.28:

$$\begin{Bmatrix} t_n \\ t_s \end{Bmatrix} = \underbrace{\begin{bmatrix} \frac{f_t}{\kappa} \exp\left(-\frac{f_t}{G_f} \kappa\right) & 0 \\ 0 & d_{\text{init}} \end{bmatrix}}_{\mathbf{S}} \begin{Bmatrix} \llbracket u \rrbracket_n \\ \llbracket u \rrbracket_s \end{Bmatrix} \quad (7.28)$$

Conforme destacado, uma fissura pode ainda se apresentar completamente fechada, indicado por  $\llbracket u \rrbracket_n < 0$ . Neste caso, emprega-se uma rigidez suficientemente alta na direção normal à fissura para evitar a interpenetração de suas superfícies.

Destaca-se que, mesmo com a introdução da fissura, a possível degradação do meio no entorno desta continua sendo avaliada através de métodos constitutivos.

A partir das Equações 7.27 e 7.28, ressalta-se que o modelo numérico proposto neste trabalho é mais adequado para abertura de trinca em modo I predominante. Em Fang et al. (2008), propõe-se um modelo em que o comportamento de amolecimento na interface é dirigido também pelo deslizamento por cisalhamento das superfícies da fissura, sendo adequado também para abertura de trinca em modo II predominante ou mesmo uma combinação dos modos I e II.

## Capítulo 8

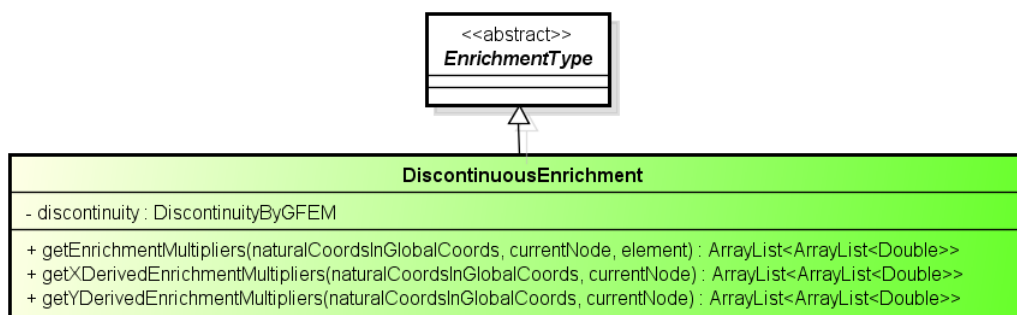
# NUCLEAÇÃO E PROPAGAÇÃO DE TRINCAS E O NÚCLEO NUMÉRICO DO INSANE

Neste capítulo, apresenta-se a estrutura implementada no núcleo numérico do sistema INSANE para modelagem de nucleação e propagação de trincas coesivas em meios bidimensionais. Conforme destacado por vezes ao longo deste texto, a presença de trincas é simulada via abordagem de enriquecimento nodal a partir do arcabouço de análise pelo MEFG desenvolvido por Alves (2012) no sistema INSANE. Sendo assim, tomou-se este arcabouço como a base da estrutura para análise de nucleação e propagação de trincas. As principais intervenções neste arcabouço, bem como em outras frentes do sistema, são destacadas ao longo deste capítulo.

Na estrutura implementada, cada trinca é uma instância da classe *DiscontinuityByGFEM*, responsável pelo seu gerenciamento. Esta classe é também responsável pelo gerenciamento de entalhes, abordados na seção C.3. No apêndice D, apresentam-se as variáveis e os métodos da classe *DiscontinuityByGFEM*.

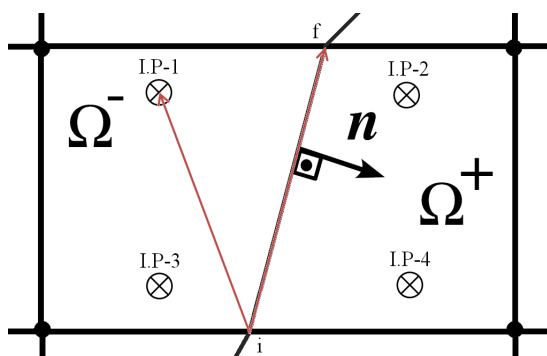
A função de enriquecimento do tipo degrau foi implementada com base na função clássica de *Heaviside*, conforme seção 5.2.2, com o nome *DiscontinuousEnrichment*. A cada instância da classe *DiscontinuityByGFEM*, tem-se associada uma instância da classe *DiscontinuousEnrichment*. Na Figura 8.1, apresenta-se o diagrama da classe *DiscontinuousEnrichment*.





**Figura 8.1:** Diagrama da classe *DiscontinuousEnrichment*.

Este enriquecimento retorna um multiplicador de valor igual à 0 ou 1 para construção da função de forma do nó enriquecido no ponto em análise, partindo-se da investigação da posição deste ponto em relação à correspondente trinca. Para esta investigação, verifica-se o sinal do produto vetorial do segmento da trinca e o vetor definido pelo ponto inicial deste segmento e o ponto em análise, conforme exemplificado na Figura 8.2 para o caso de investigação do valor deste multiplicador em um dos pontos de integração do elemento cortado pela trinca.



**Figura 8.2:** Exemplo de investigação do valor do multiplicador de enriquecimento.

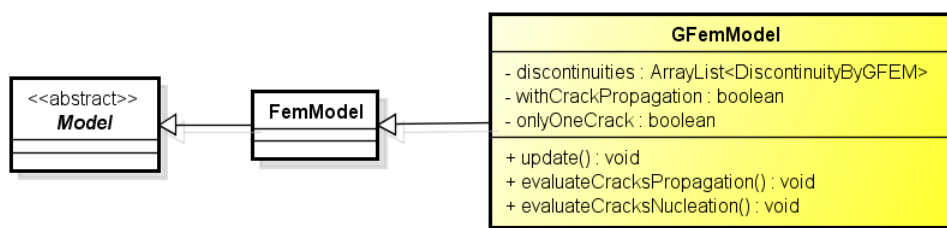
Como segmento da trinca, entende-se o trecho da trinca contido no elemento finito. Neste trabalho, os segmentos da trinca são inseridos como linhas retas de aresta a aresta do elemento. Conforme Wolff (2010), uma vez que, no contexto teórico de fissura coesiva, a energia total de fratura é dissipada com a separação entre as superfícies da fissura e não com a criação destas superfícies, o resultado numérico do método aqui descrito não é particularmente sensível a quando exatamente uma trinca é estendida, ou mesmo a qual o valor do comprimento desta extensão.

Os pontos inicial e final de um segmento da trinca são classificados como tal em função do sentido da trinca, por sua vez definido pela classe *DiscontinuityByGFEM*. Desta forma, transfere-se aos segmentos o conhecimento relativo à definição dos subdomínios “negativo” e “positivo” do corpo em relação à trinca, para correta determinação do multiplicador de enriquecimento.

Em Alves (2012), encontram-se maiores detalhes sobre a lógica empregada no sistema INSANE para construção das funções de forma a partir de enriquecimento nodal, implementada na classe *EnrichedShape*.

Na classe *GFemModel*, incluíram-se as variáveis *discontinuities*, *withCrackPropagation* e *onlyOneCrack*. A variável *discontinuities* consiste em uma lista de armazenamento de objetos *DiscontinuityByGFEM*, ou seja, é responsável por armazenar as trincas, que são identificadas pelo núcleo numérico durante a análise, bem como os entalhes, que por sua vez são parâmetros de entrada. As variáveis *withCrackPropagation* e *onlyOneCrack* consistem em booleanos que definem, respectivamente, se a análise deve avaliar o processo de nucleação e propagação de trincas e, no caso afirmativo, se somente uma trinca deve ser nucleada.

Na Figura 8.3, resumem-se as intervenções na classe *GFemModel*, a partir do seu diagrama de classe.



**Figura 8.3:** Intervenções na classe *GFemModel*.

Para uma melhor compreensão e acompanhamento da continuidade deste capítulo, na Figura 8.4, apresenta-se o fluxograma do processo de nucleação e propagação de trincas implementado no sistema INSANE.

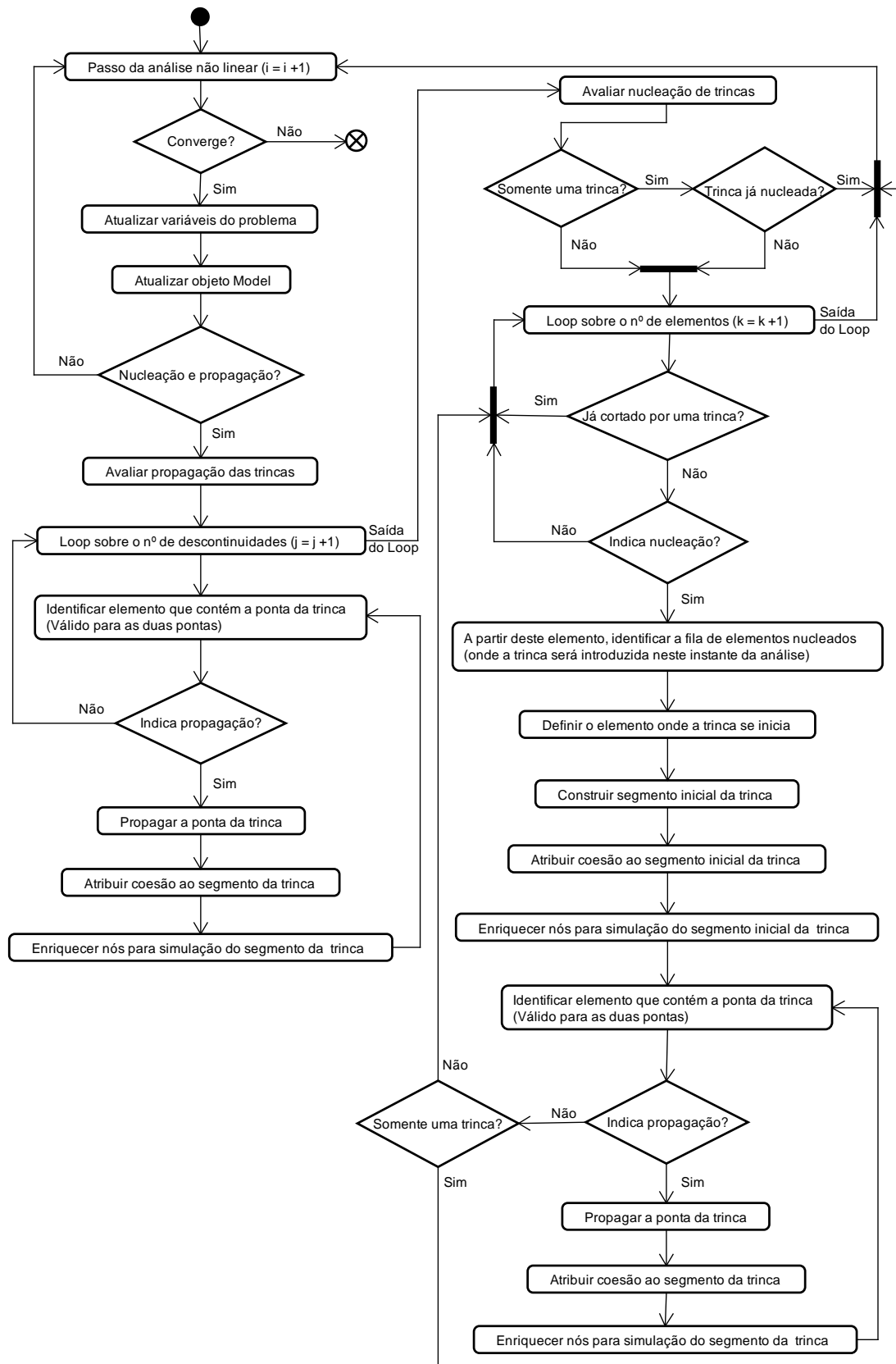


Figura 8.4: Fluxograma do processo de nucleação e propagação de trincas.

Neste trabalho, a verificação do fenômeno de nucleação, bem como da propagação de trincas, somente é realizada em um sistema equilibrado, ou seja, na convergência de um passo da análise.

No sistema INSANE, ao se alcançar a convergência de um passo, atualizam-se as variáveis do problema. Dentre estas variáveis, tem-se o objeto *Model*. No caso de análise pelo MEFG, este objeto consiste em uma instância da classe *GFemModel*, que, para sua atualização, utilizava-se do método *update* da superclasse *FemModel*.

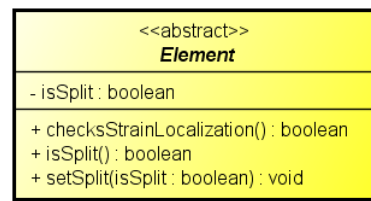
Neste trabalho, sobrepôs-se em *GFemModel* o método *update* da classe *FemModel*, conforme destacado na Figura 8.3. O método implementado, além de acionar o método *update* de *FemModel*, é responsável por disparar os métodos de propagação das trincas e de nucleação de fissuras, nesta ordem, se *withCrackPropagation* for afirmativo.

Os métodos de propagação das trincas e de nucleação de fissuras foram implementados em *GFemModel* com os nomes *evaluateCracksPropagation* e *evaluateCracksNucleation*, respectivamente, conforme apresentado na Figura 8.3. Se *onlyOneCrack* for afirmativo, *evaluateCracksNucleation* será disparado somente até se nuclear a primeira, e única, trinca.

Conforme observado, permite-se a nucleação de mais de uma trinca na análise. Esta possibilidade é a razão de se estabelecer uma medida comum para nucleação e para propagação das trincas. Medidas diferentes poderiam resultar em uma trinca nucleando em um elemento, onde na verdade se deveria ter a propagação de uma trinca já existente. Esta é também a razão de se primeiro verificar a propagação das trincas existentes e, posteriormente, a nucleação de fissuras.

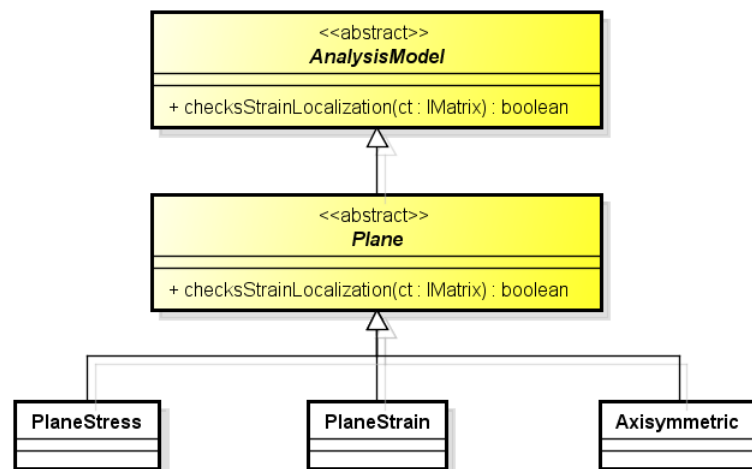
No método *evaluateCracksNucleation*, verifica-se o fenômeno de nucleação em elemento por elemento da malha, com exceção daqueles já cortados por uma ou mais trincas. Para tanto, dispara-se o método *checksStrainLocalization* implementado na classe *Element*. Neste método, tem-se duas possibilidades de verificação, definida em função do meio material.

Se o meio for linear elástico, condicionado ao material *ConcreteCohesiveCrack* implementado por Wolff (2010) no sistema INSANE, o fenômeno de nucleação é indicado se em algum ponto de Gauss a máxima tensão principal for maior ou igual à resistência à tração do material, ao passo que, se não linear, este fenômeno é indicado se em algum ponto de Gauss o *tensor acústico* for singular para algum vetor unitário, conforme discutido no capítulo 6. Na Figura 8.5, apresentam-se as intervenções na classe *Element*.



**Figura 8.5:** Intervenções na classe *Element*.

Conforme destacado no capítulo 6, por ser especializado para análise plana, o método empregado neste trabalho para verificação da existência de algum vetor unitário que singularize o *tensor acústico* foi implementado na classe modelo de análise *Plane*. Este método também se chama *checksStrainLocalization* e recebe como argumento o tensor de rigidez tangente. Na Figura 8.6, apresentam-se as intervenções nas classes *AnalysisModel* e *Plane*, a partir do diagrama de classe.



**Figura 8.6:** Intervenções nas classes *AnalysisModel* e *Plane*.

A partir da identificação do fenômeno de nucleação, cria-se uma instância da classe *DiscontinuityByGFEM* através do construtor específico para trincas. Este construtor recebe como argumentos um nome para a trinca, definido pelo programa, o objeto *Model* e o elemento onde se identificou este fenômeno. A partir deste construtor, dispara-se o método *buildCrack* repassando-se o elemento.

Em função do tamanho do passo da análise não linear e do refinamento da malha, é possível que o sistema equilibrado resulte na introdução de uma trinca em uma fila de elementos simultaneamente, ao invés de em somente um elemento. Desta forma, o elemento onde se identificou o fenômeno de nucleação pode na verdade fazer parte de uma fila de elementos nucleados, não sendo o único onde a trinca deverá ser introduzida neste instante. Sendo assim, o método *buildCrack* se inicia buscando-se esta fila de elementos. Para tanto, parte-se da determinação de uma possível direção para o segmento da trinca que cortará o elemento já identificado.

Esta direção é definida como sendo a direção normal à máxima deformação principal não local, calculada no centroide do elemento. Para este caso, o tensor de deformações não local é calculado como a média ponderada das deformações nos pontos de Gauss do elemento a partir da função de peso Gaussiana definida na Equação 8.1:

$$w = \frac{1}{(2\pi)^{3/2}l^3} \exp\left(-\frac{r^2}{2l^2}\right) \quad (8.1)$$

onde,

$w$  é o valor da função de peso Gaussiana em um ponto de Gauss;

$l$  determina quão rapidamente a função de peso diminui longe do ponto em análise;

$r$  é a distância de um ponto de Gauss ao ponto em análise.

Segundo Dumstorff e Meschke (2007), o parâmetro  $l$  que determina quão rapidamente a função de peso diminui longe do ponto em análise pode ser assumido como sendo 1,5 vezes o comprimento interno do elemento  $l_c$ , por sua vez definido conforme Equação 8.2:

$$l_c = \sqrt{A_e} \quad (8.2)$$

onde,

$A_e$  é a área média de um elemento finito.

Este método foi implementado sob o nome *getInitialSegmentDirectionByNonLocalPrincipalStrain* na classe *DiscontinuityByGFEM*. Recebe como argumentos o elemento e o seu centroide.

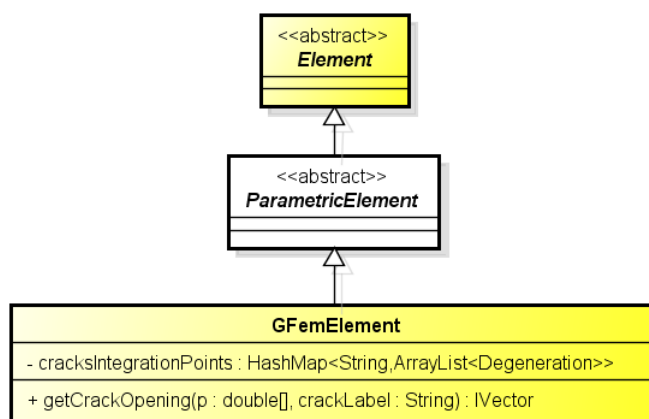
A partir da determinação desta possível direção para o segmento da trinca, identificam-se os possíveis dois elementos adjacentes ao elemento já identificado que contêm os extremos deste segmento. Para tanto, dispõe-se este segmento passando pelo centroide do elemento.

Em seguida, verifica-se o fenômeno de nucleação nestes elementos através do método *checksStrainLocalization* da classe *Element*, abordado anteriormente. Se algum destes elementos indicar a nucleação, repete-se o processo descrito verificando-se a nucleação no possível elemento adjacente a este que interage com a trinca e assim sucessivamente. Ao final desta busca, determinam-se os elementos de extremo desta fila e assume-se como o inicial aquele que apresentar a maior deformação principal, avaliada em seus pontos de Gauss.

Esta lógica evita que a análise seja dependente do primeiro elemento identificado pertencente a uma fila de elementos nucleados.

Para o elemento inicial, assume-se que a disposição e a direção do segmento da trinca são aquelas determinadas durante a definição da fila de elementos, ou seja, assume-se que o segmento da trinca no elemento inicial passa em seu centroide e tem a direção definida como sendo a direção normal à máxima deformação principal não local, calculada neste ponto. Ao se definir a disposição e a direção de um segmento da trinca, armazenam-se no correspondente elemento os pontos inicial e final deste segmento. Quanto aos demais elementos identificados, desprezam-se as considerações feitas, tratando-os posteriormente.

Na sequência, para atribuição da coesão ao segmento inicial da trinca, o método *buildCrack* dispara o método *setsCohesion* da classe *DiscontinuityByGFEM* passando o elemento inicial como argumento. Neste método, definem-se os parâmetros do modelo constitutivo discreto, armazenando-os em pontos de integração dispostos ao longo do segmento da trinca. Para tanto, inicia-se determinando o número de pontos necessários para a integração numérica exata da parcela da matriz de rigidez relativa à lei coesiva, observada na Equação 7.19. Este número é definido em função do grau do polinômio interpolador do elemento. Para armazenamento destes pontos de integração, incluiu-se na classe *GFemElement* a variável *cracksIntegrationPoints*. Esta variável consiste em um mapa de armazenamento de listas de pontos de integração, cuja chave consiste no nome atribuído à trinca. Na Figura 8.7, apresentam-se as intervenções na classe *GFemElement*, a partir do seu diagrama de classe.



**Figura 8.7:** Intervenções na classe *GFemElement*.

Em seguida, definem-se os parâmetros do modelo constitutivo discreto. Conforme capítulo 7, o modelo constitutivo discreto empregado neste trabalho foi formulado para uma lei coesiva particular. Este modelo foi implementado por Wolff (2010) no sistema INSANE sob o nome *CohesiveCrackConstModel*. Conforme destacado, a lei coesiva adotada requer a determinação de 4 parâmetros, a saber: resistência à tração, energia de fratura, rigidez inicial ao cisalhamento e rigidez à interpenetração das superfícies da fissura.



Com exceção do modelo constitutivo linear elástico, condicionado ao material *ConcreteCohesiveCrack*, os modelos constitutivos distribuídos não possuem todos estes parâmetros de entrada. Para preservar a generalidade do sistema, optou-se por recorrer a artifícios para obtenção destes parâmetros, em suas ausências, ao invés de adequar estes modelos.

Para a resistência à tração, parte-se da Equação 7.23. Observa-se que, para a abertura histórica da trinca igual à zero, tem-se o componente normal da lei coesiva igual à resistência à tração. Isto se deve ao fato de que esta equação foi proposta para análise de propagação de trincas onde a medida de propagação é a resistência à tração, ou seja, no instante da introdução do segmento da trinca, a tensão normal a este segmento é igual a esta resistência. Neste trabalho, associar a resistência à tração à abertura histórica da trinca igual à zero gera instabilidade numérica, tendo em vista que, no instante da introdução do segmento da trinca, a tensão normal a este segmento não é, obrigatoriamente, igual a esta resistência. Desta forma, contorna-se esta instabilidade e satisfaz-se a generalidade assumindo-se como pseudo  $f_t$  a tensão normal ao segmento da trinca no instante de sua introdução, tal que se tenha uma amarração consistente entre os modelos constitutivos distribuído e discreto.

Para a energia de fratura, na ausência desta informação na lei material, assume-se que a energia total a ser dissipada com a separação entre as superfícies do segmento da trinca é aquela contida no elemento no instante da nucleação de fissuras, ou seja, assume-se como pseudo  $G_f$  a energia de deformação do elemento dividida pela área do segmento de trinca.

A rigidez inicial ao cisalhamento é definida em função do módulo de elasticidade transversal e do fator de retenção ao cisalhamento, ao passo que a rigidez à interpenetração das superfícies da fissura em função do módulo de elasticidade.

Definida a coesão, o método *buildCrack* continua a construção da trinca definindo os nós do elemento inicial a serem enriquecidos. Para tanto, verifica-se se as arestas que contêm as pontas do segmento inicial da trinca são internas ou externas.

Nós pertencentes à aresta interna que contém a ponta da trinca não são enriquecidos. A partir do critério da continuidade do MEF, constata-se que, desta forma, simula-se a abertura nula de uma ponta de trinca.

Na sequência, o método *buildCrack* dispara o método *enrichingNodes* da classe *DiscontinuityByGFEM* passando como argumento a lista de nós a serem enriquecidos. Neste método, atribui-se aos nós a função de enriquecimento do tipo degrau associada à descontinuidade, bem como adequam-se as grandezas nodais aos graus de liberdade adicionais. Quanto às restrições nodais, assume-se que, se um grau de liberdade convencional for restrito, os graus de liberdade adicionais associados a este também o serão. Esta é uma opção para se respeitar as condições de contorno, tendo em vista que os deslocamentos nodais passam a ser funções também dos graus de liberdade adicionais relativos ao enriquecimento do tipo degrau.

Para finalizar a construção da trinca, retorna-se aos demais elementos nucleados. Para tanto, o método *buildCrack* dispara o método *update* da classe *DiscontinuityByGFEM*, o mesmo adotado para propagação das trincas.

Neste método, verifica-se a propagação das duas pontas da trinca. Para tanto, identificam-se os possíveis dois elementos que contêm estas pontas. Na sequência, verifica-se o fenômeno de nucleação nestes elementos através do método *checksStrainLocalization* da classe *Element*.

Se algum destes elementos indicar a nucleação, primeiramente, define-se a direção do segmento da trinca que o cortará de maneira análoga ao realizado na definição da fila de elementos nucleados.

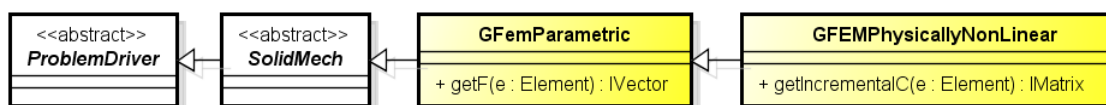
No entanto, neste caso, o tensor de deformações não local é calculado na ponta da trinca, ao passo que, tal como em Wolff (2010), para a média ponderada consideram-se todos os pontos de Gauss compreendidos em um círculo centralizado nesta ponta e de raio igual à 2,0 vezes o comprimento interno do elemento  $l_c$ , por sua vez definido conforme Equação 8.2. Ressalta-se que, neste trabalho, limita-se à 30 graus a variação da direção de um segmento da trinca em relação ao anterior.

Este método foi implementado sob o nome *getDiscontinuityDirectionByNonLocalPrincipalStrain* na classe *DiscontinuityByGFEM*. Recebe como argumentos o elemento, a ponta da trinca e o ponto anterior.

Por fim, dispõe-se este segmento a partir da ponta a se propagar e definem-se a coesão e o enriquecimento nodal, tal como para o segmento inicial da trinca. Este processo se repete até que se depare com um elemento em que não se indique a nucleação ou com uma aresta externa.

Quanto ao método *evaluateCracksPropagation*, verifica a propagação de trinca por trinca do modelo. Para tanto, percorre-se a lista *discontinuities* disparando-se o referido método *update*. Sendo assim, ressalta-se outra aplicação deste sistema, que consiste na análise de propagação de uma falha preexistente. Para tanto, define-se esta falha como um entalhe e opta-se por *onlyOneCrack*. Para restringir a nucleação a uma única trinca, condiciona-se esta verificação à lista *discontinuities* vazia. Recordando-se que esta lista também armazena os entalhes, fica evidente esta aplicação. Ressalva-se que, neste caso, a coesão será atribuída somente à extensão da falha preexistente, por sua vez tratada como um corte sem mecanismos de tenacidade.

Finalizaram-se as intervenções no sistema INSANE para inserção desta estrutura com as adequações dos métodos *getF* e *getIncrementalC* das classes *GFemParametric* e *GFEMPhysicallyNonLinear*, respectivamente. Estes métodos foram adaptados para recorrerem ao mapa de armazenamento dos pontos de integração dos segmentos de trincas contidos no elemento, tal que o vetor de forças equivalentes às tensões internas e a matriz de rigidez do elemento passaram a considerar a contribuição da parcela relativa à lei coesiva, conforme Equações 7.16 e 7.19, respectivamente. Na Figura 8.8, apresentam-se estas intervenções.



**Figura 8.8:** Intervenções nas classes *GFemParametric* e *GFEMPhysicallyNonLinear*.

## Capítulo 9

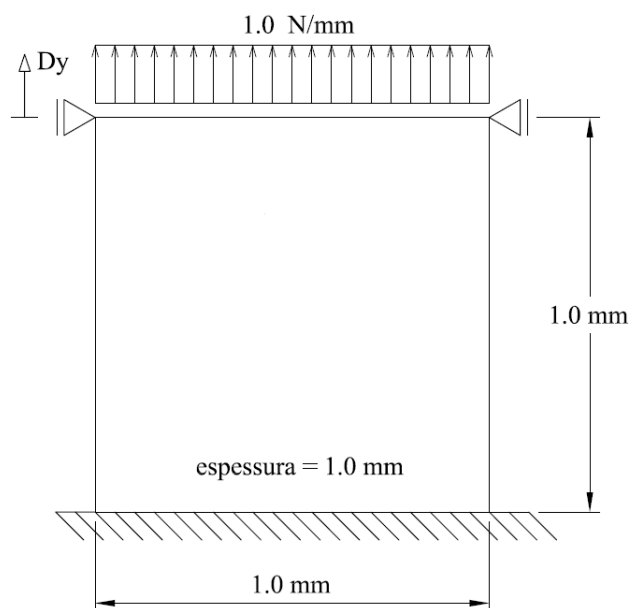
# SIMULAÇÕES NUMÉRICAS

Neste capítulo, apresentam-se 6 simulações numéricas para avaliar o desempenho e a generalidade do modelo numérico implementado, listadas a seguir. No sistema gráfico integrado, para análise de nucleação e propagação de trincas, seleciona-se esta opção no diálogo de definição do modelo constitutivo, ilustrado na Figura C.14. Neste diálogo, define-se também se somente uma trinca deve ser nucleada na análise.

- Seção 9.1: Tração direta - Objetivou-se demonstrar o emprego deste sistema para análise de nucleação e propagação de trincas em meio linear elástico;
- Seção 9.2: Flexão em 3 pontos - Objetivou-se demonstrar o emprego deste sistema para análise de nucleação e propagação de trincas adotando-se modelo constitutivo de fissuras distribuídas;
- Seção 9.3: Painel em “L” - Objetivou-se demonstrar o emprego deste sistema para análise de nucleação e propagação de trincas adotando-se modelo de dano;
- Seção 9.4: Compressão diametral - Objetivou-se demonstrar o emprego deste sistema para análise de nucleação e propagação de múltiplas trincas;
- Seção 9.5: Viga em balanço - Objetivou-se demonstrar o desempenho deste sistema na análise de nucleação e propagação de trincas em elementos onde as condições de contorno naturais e essenciais são prescritas;
- Seção 9.6: Cisalhamento em 4 pontos - Objetivou-se demonstrar o emprego deste sistema para análise de propagação de falha preexistente.

## 9.1 Tração Direta

A chapa sob tração direta, cuja configuração é apresentada na Figura 9.1, foi simulada a partir do sistema gráfico integrado objetivando-se demonstrar o emprego deste sistema para análise de nucleação e propagação de trincas em meio linear elástico.

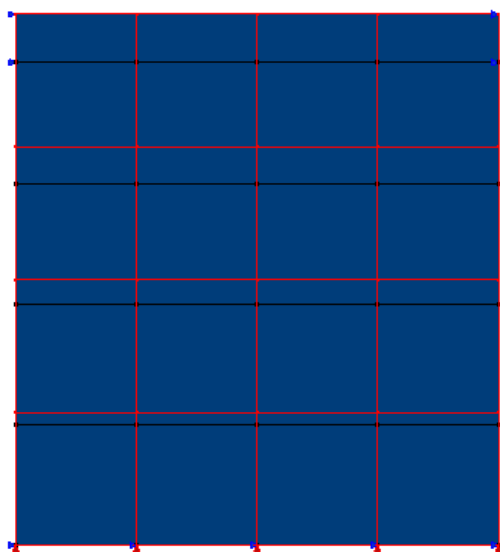


**Figura 9.1:** Tração direta: Configuração (Wolff, 2010).

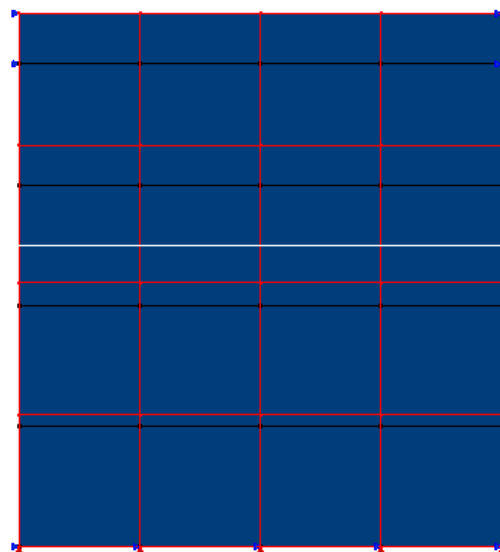
Nesta análise, adotou-se o modelo constitutivo linear elástico, cujos parâmetros do material são: Módulo de elasticidade  $E = 100 \text{ N/mm}^2$ , coeficiente de Poisson  $\nu = 0,0$ , resistência à tração  $f_t = 1,0 \text{ N/mm}^2$ , energia de fratura  $G_f = 0,02 \text{ N/mm}$  e rigidez inicial ao cisalhamento  $d_{init} = 0,0 \text{ N/mm}^3$ . A chapa foi modelada com elementos quadrilaterais de quatro nós  $Q4$  em estado plano de tensão.

Para solução do modelo, empregou-se o método de controle direto de deslocamento, incrementando-se de  $0,0005 \text{ mm}$  o deslocamento vertical  $D_y$ , com tolerância para convergência de  $1 \times 10^{-3}$  e carga de referência igual à  $1,0 \text{ N/mm}$ .

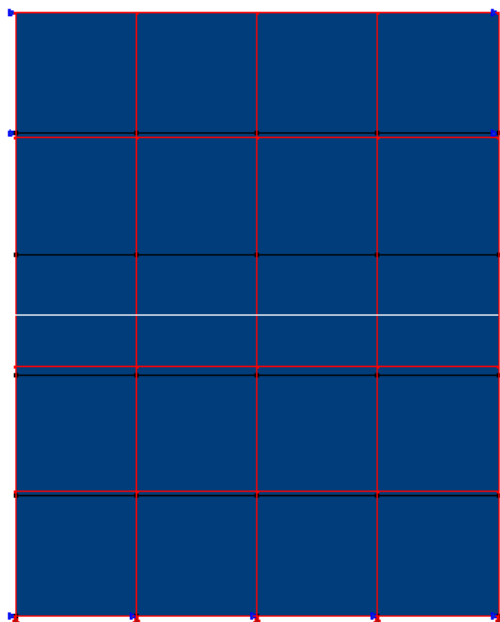
Nas Figuras 9.2 e 9.3, apresentam-se, respectivamente, a deformada com fator de escala igual a 10 e a tensão normal vertical  $\sigma_{yy}$  ao longo do domínio, para os passos 20 ( $1,0 \text{ N/mm}$ ), 21 ( $0,95 \text{ N/mm}$ ), 50 ( $0,34 \text{ N/mm}$ ) e 100 ( $0,086 \text{ N/mm}$ ).



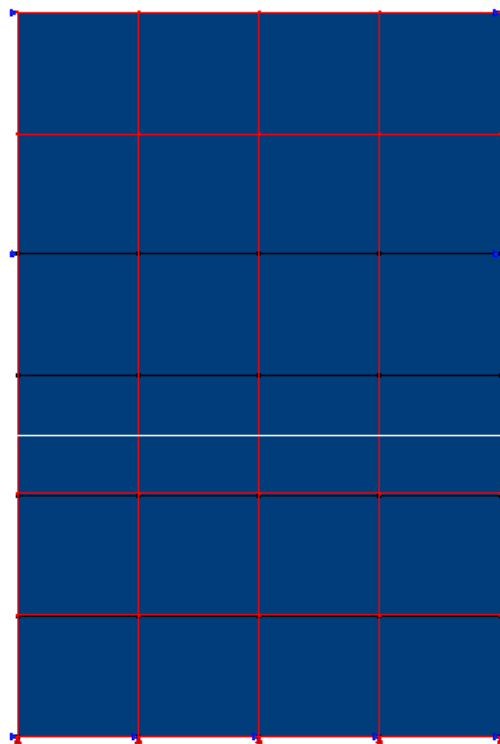
Passo 20 (1,0 N/mm)



Passo 21 (0,95 N/mm)

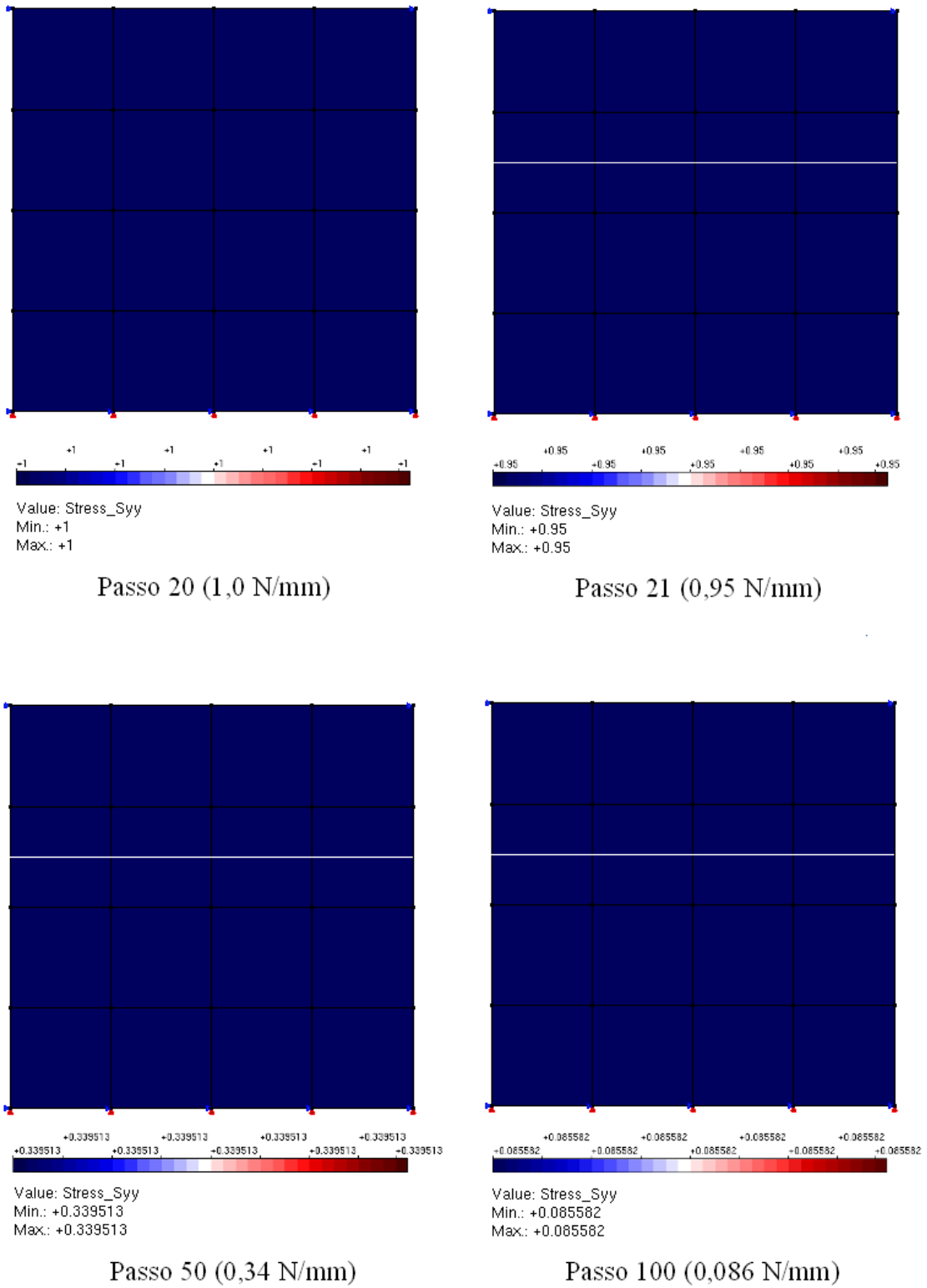


Passo 50 (0,34 N/mm)



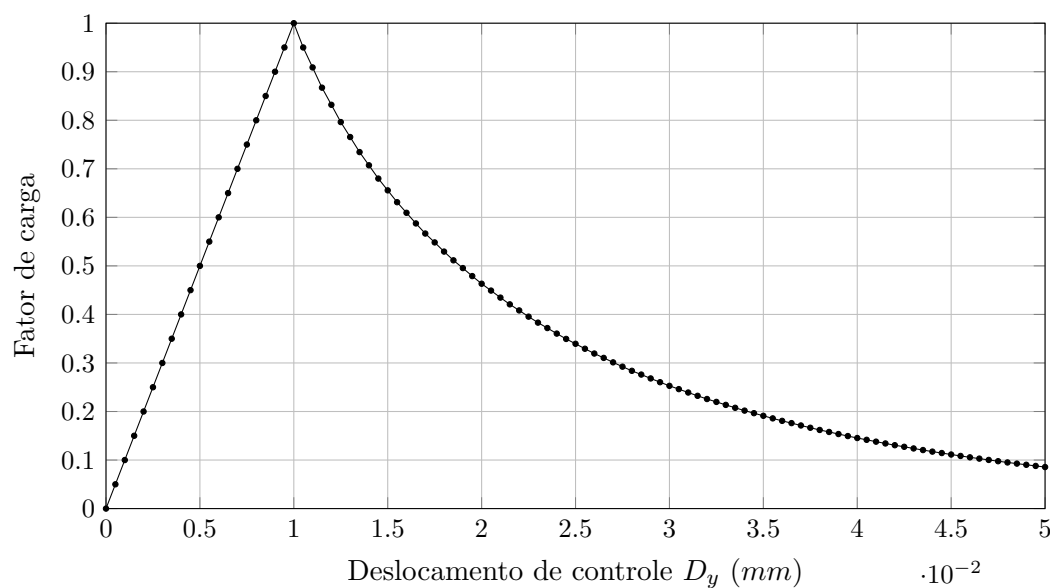
Passo 100 (0,086 N/mm)

**Figura 9.2:** Tração direta: Deformada.



**Figura 9.3:** Tração direta: Tensão  $\sigma_{yy}$ .

A partir das Figuras 9.2 e 9.3, observa-se que, conforme esperado, a trinca surge no passo em que a chapa se encontra em um estado de tensão igual à sua resistência à tração, cortando simultaneamente todos os elementos pertencentes à linha de elementos nucleados. Neste instante, o modelo constitutivo discreto, responsável pela não linearidade do processo, passa a comandar a análise. Na Figura 9.4, apresenta-se a trajetória de equilíbrio do deslocamento de controle  $D_y$ .



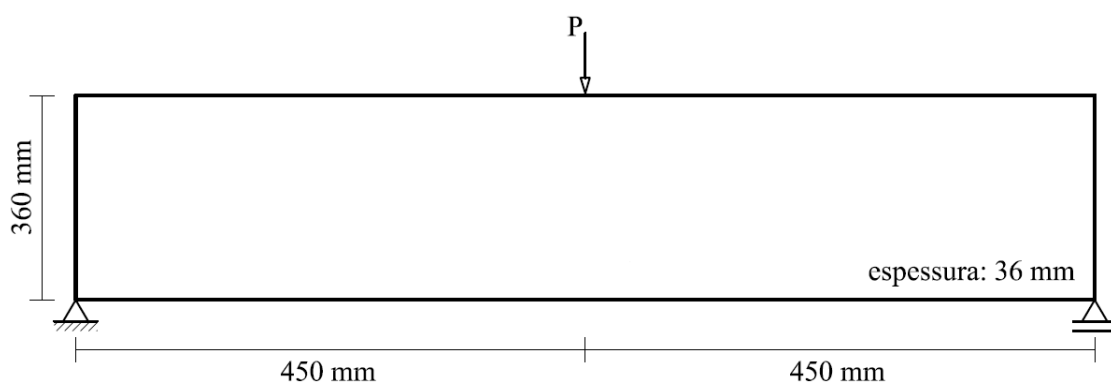
**Figura 9.4:** Tração direta: Trajetória de equilíbrio do deslocamento de controle  $D_y$ .

Nesta análise, optou-se por somente uma trinca. Caso contrário, surgiria uma trinca para cada linha de elementos, tendo em vista o estado de tensão uniforme ao longo do domínio. Este exemplo, além de destacar o emprego do sistema para análise de nucleação e propagação de trincas em meio linear elástico, exalta o funcionamento do modelo constitutivo discreto. Os resultados obtidos nesta modelagem coincidem com os resultados publicados por Mergheim et al. (2005), onde se objetivou verificar a influência de diferentes discretizações na relação carga-deslocamento no modelo de fissuras coesivas.



## 9.2 Flexão em 3 Pontos

A viga sob flexão em 3 pontos, cuja configuração é apresentada na Figura 9.5, foi simulada a partir do sistema gráfico integrado objetivando-se demonstrar o emprego deste sistema para análise de nucleação e propagação de trincas adotando-se modelo constitutivo de fissuras distribuídas.

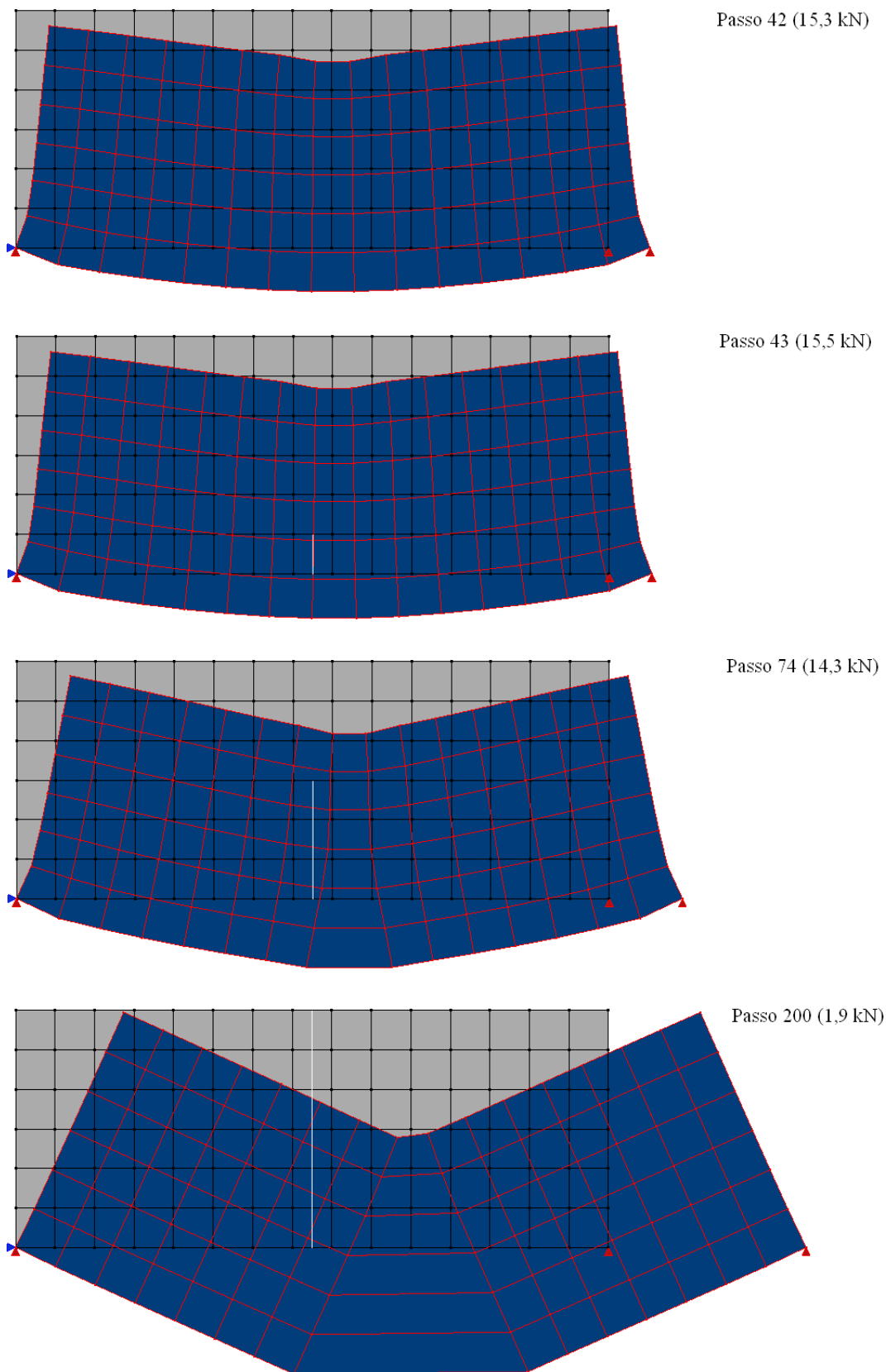


**Figura 9.5:** Flexão em 3 pontos: Configuração (Penna, 2011).

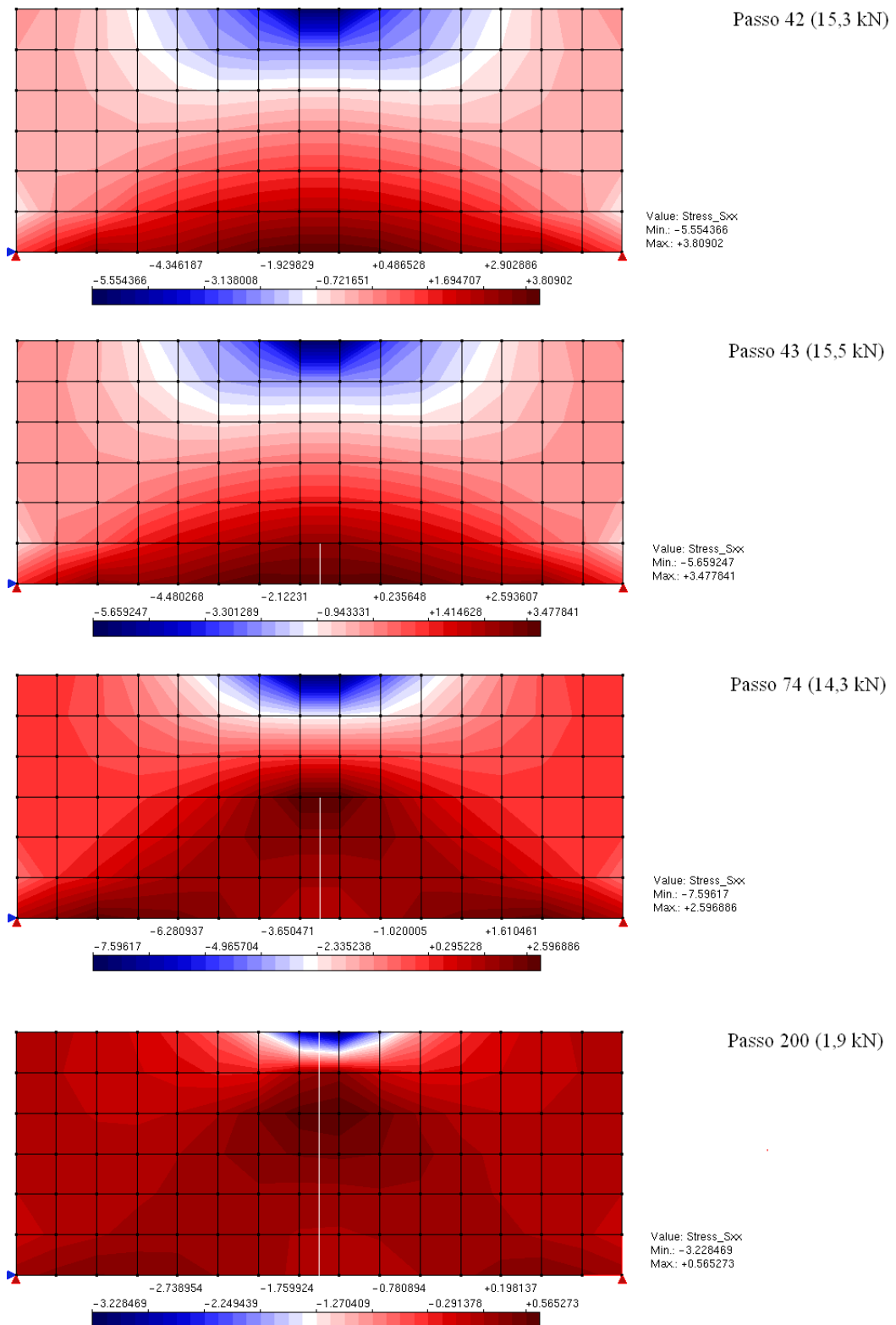
Nesta análise, adotou-se o modelo constitutivo de fissuras distribuídas com leis de Carreira e Chu (1985), para compressão, e Boone et al. (1986), para tração, cujos parâmetros do material são: Módulo de elasticidade elástico  $E_0 = 44000,0 \text{ N/mm}^2$ , coeficiente de Poisson  $\nu = 0,2$ , resistência à compressão  $f_c = 40,0 \text{ N/mm}^2$ , resistência à tração  $f_t = 3,8 \text{ N/mm}^2$ , deformação relativa ao limite elástico na compressão  $\varepsilon_c = 0,0018$ , energia de fratura  $G_f = 0,164 \text{ N/mm}$ , comprimento característico  $h = 100,0 \text{ mm}$  e fator de retenção ao cisalhamento  $\beta_r = 0,05$ . A viga foi modelada com elementos quadrilaterais de quatro nós  $Q4$  em estado plano de tensão.

Para solução do modelo, empregou-se o método de controle direto de deslocamento, incrementando-se de  $0,0015 \text{ mm}$  o deslocamento do apoio direito, com tolerância para convergência de  $1 \times 10^{-4}$  e carga de referência  $P = 1,0 \text{ N}$ .

Nas Figuras 9.6 e 9.7, apresentam-se, respectivamente, a deformada com fator de escala igual a 1000 e a tensão normal horizontal  $\sigma_{xx}$  ao longo do domínio, para os passos 42 ( $15,3 \text{ kN}$ ), 43 ( $15,5 \text{ kN}$ ), 74 ( $14,3 \text{ kN}$ ) e 200 ( $1,9 \text{ kN}$ ).

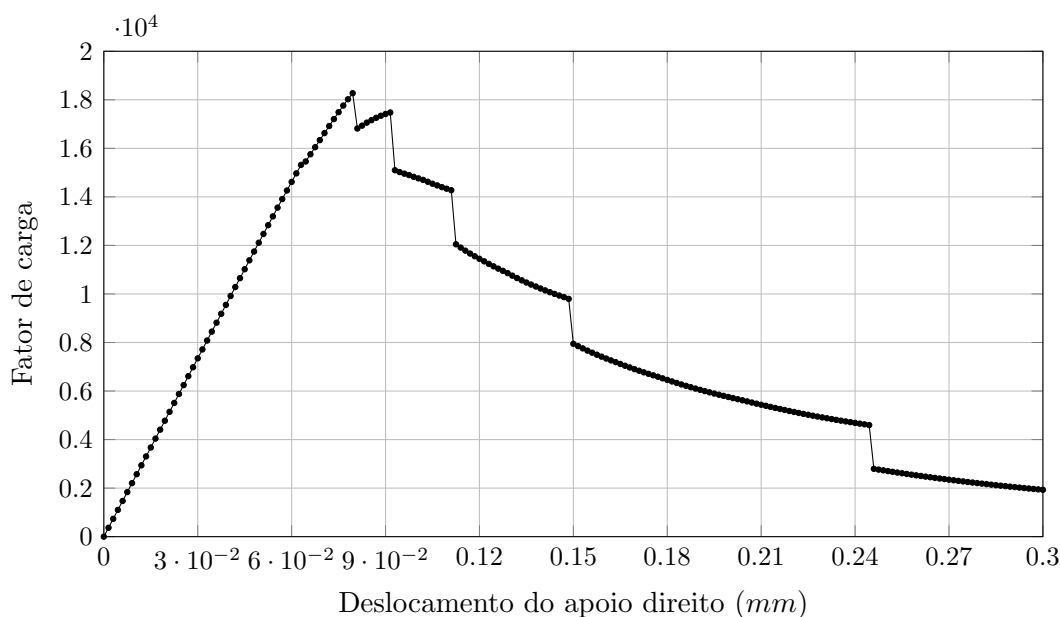


**Figura 9.6:** Flexão em 3 pontos: Deformada.



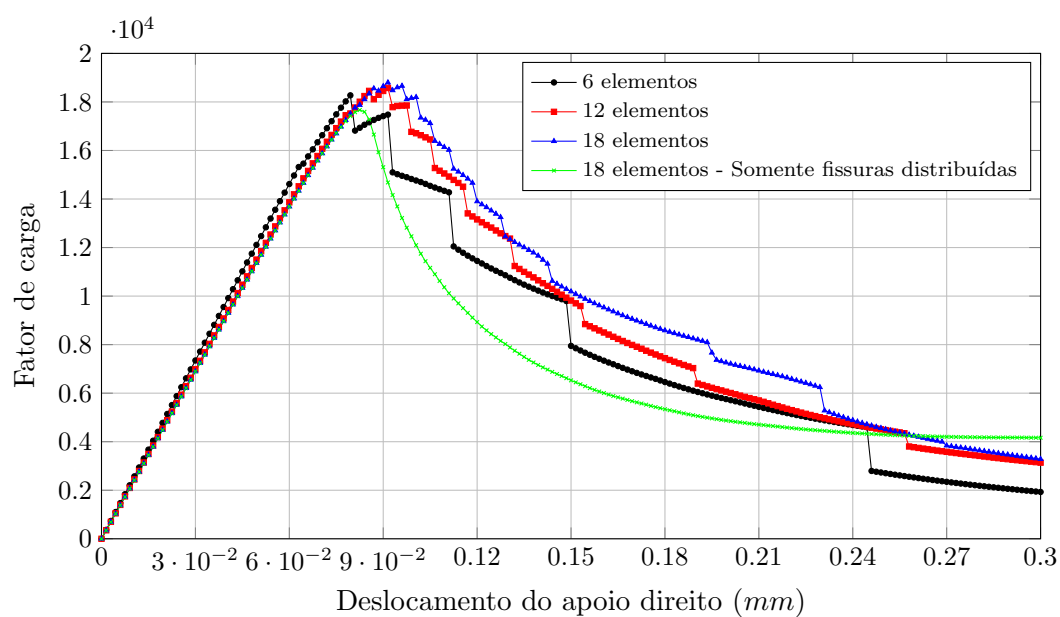
**Figura 9.7:** Flexão em 3 pontos: Tensão  $\sigma_{xx}$ .

A partir das Figuras 9.6 e 9.7, é possível observar o estágio de localização de tensões no instante que antecede a introdução da trinca, bem como o caminhamento das tensões à ponta da trinca nos instantes que antecedem sua propagação. Na Figura 9.8, apresenta-se a trajetória de equilíbrio do deslocamento do apoio direito.



**Figura 9.8:** Flexão em 3 pontos: Trajetória de equilíbrio do deslocamento do apoio direito.

Na Figura 9.8, observam-se degraus nos passos em que se introduziram segmentos da trinca. Isto demonstra que a consideração feita acerca da sensibilidade do modelo numérico aqui proposto a quando exatamente uma trinca é estendida, ou mesmo a qual o valor do comprimento desta extensão, não é tão verdade. Os resultados se mostraram sensíveis ao comprimento do segmento da trinca, ou seja, observou-se dependência de malha. Estes degraus são reduzidos à medida que se aumenta o refinamento da malha, conforme observado na Figura 9.9, onde se comparam esta trajetória de equilíbrio com as obtidas duplicando-se e triplicando-se o número de elementos ao longo da altura. Malhas grosseiras implicam na introdução precoce de trechos da trinca, tal que se tenha uma atribuição abrupta de fragilidade ao sistema estrutural.

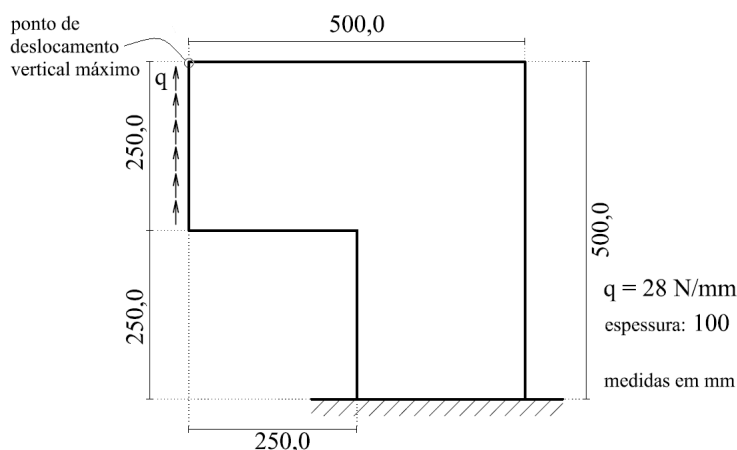


**Figura 9.9:** Flexão em 3 pontos: Comparação dos resultados obtidos com 6, 12 e 18 elementos ao longo da altura.

Na Figura 9.9, comparam-se também os resultados obtidos com o da modelagem da fissuração somente de forma distribuída. Observa-se que o modelo combinado resultou em um comportamento estrutural mais dúctil do que o modelo distribuído, o que é atribuído à coesão nas superfícies da trinca, além de apresentar uma carga de pico um pouco mais elevada.

### 9.3 Painel em “L”

O painel em “L”, cuja configuração é apresentada na Figura 9.10, foi simulado a partir do sistema gráfico integrado objetivando-se demonstrar o emprego deste sistema para análise de nucleação e propagação de trincas adotando-se modelo de dano.

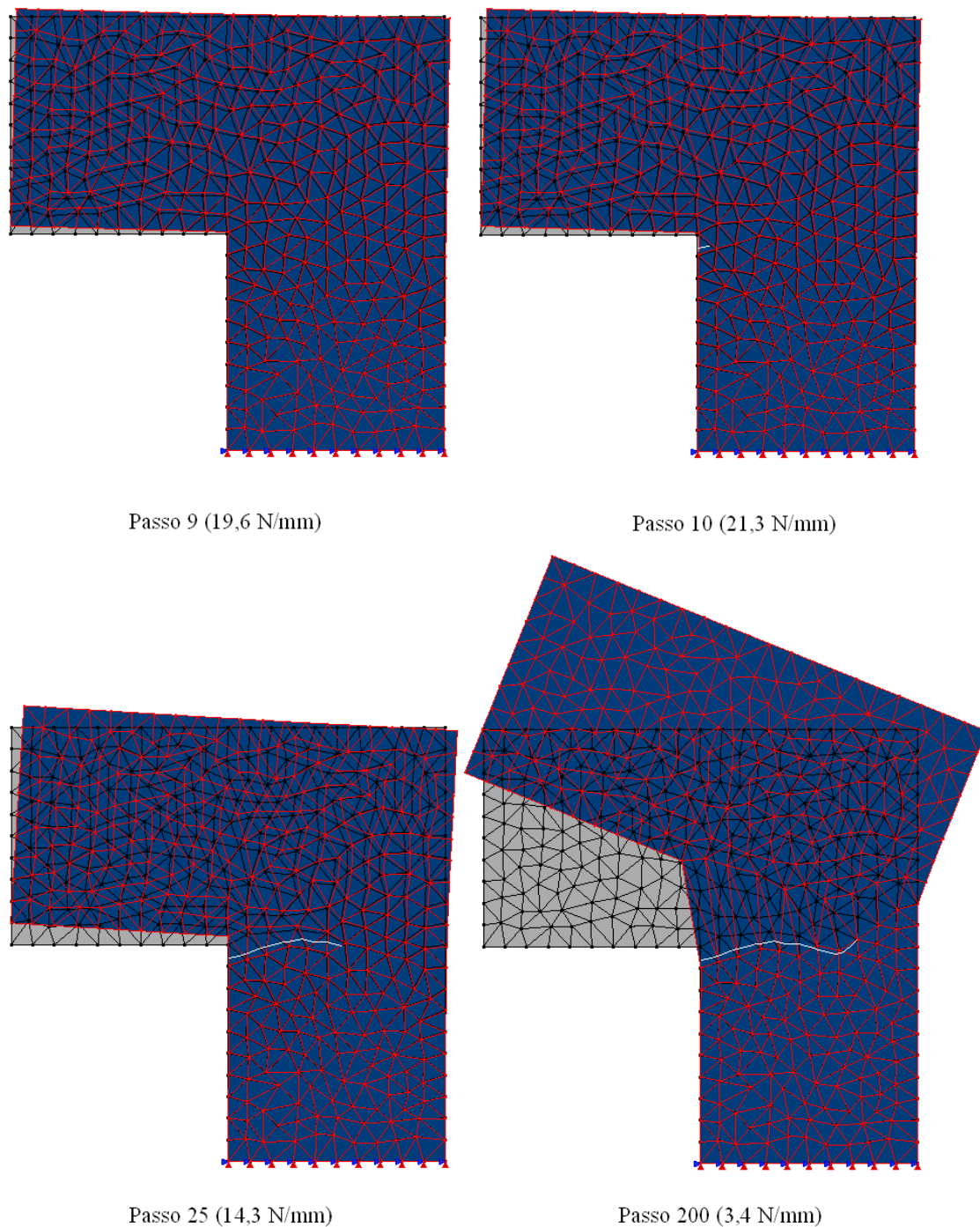


**Figura 9.10:** Painel em “L”: Configuração (Winkler et al., 2004).

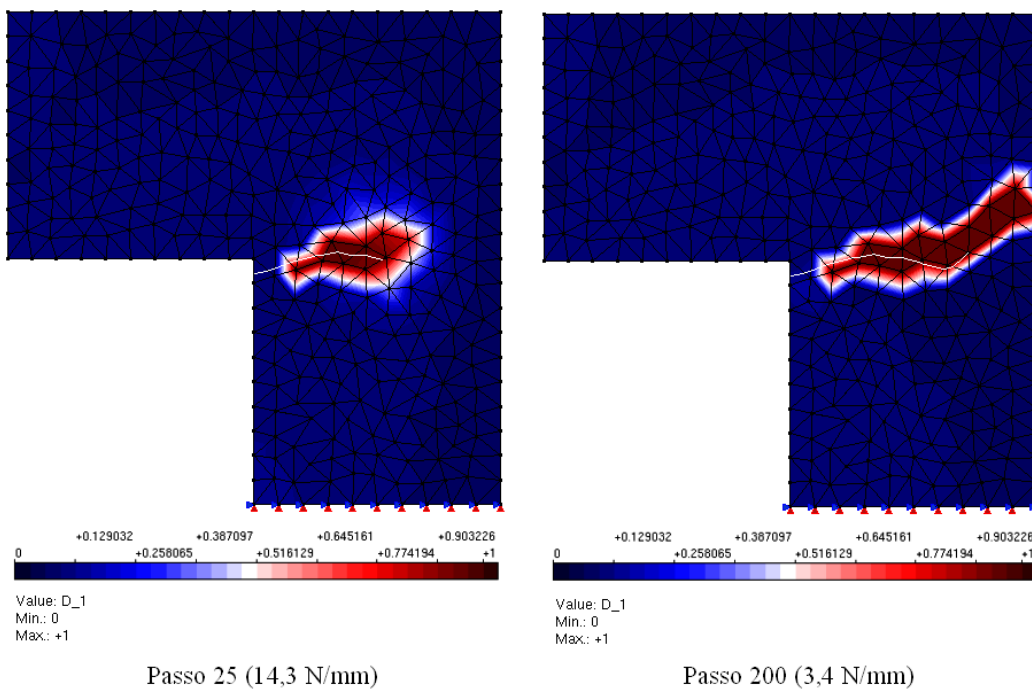
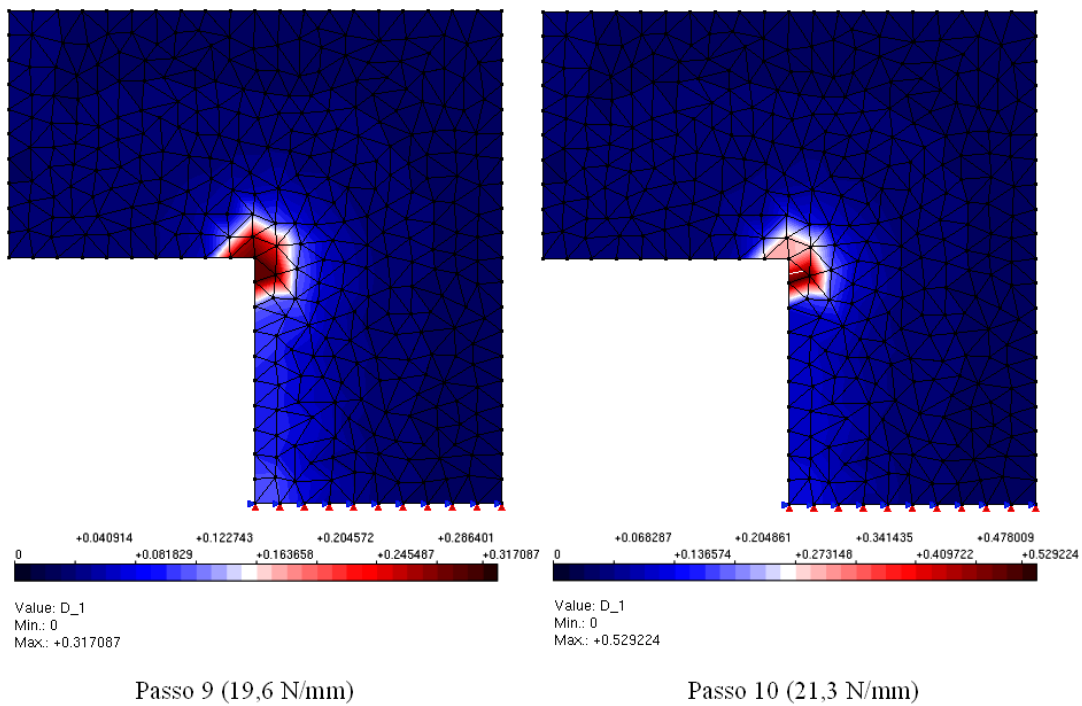
Nesta análise, adotou-se o modelo constitutivo de dano volumétrico por Penna (2011), cujos parâmetros do material são: Módulo de elasticidade elástico  $E_0 = 25850,0 \text{ N/mm}^2$  e coeficiente de Poisson  $\nu = 0,18$ . Para acompanhamento da evolução do dano, adotaram-se leis polinomiais para tração e para compressão. A lei de evolução de dano para tração é descrita pelos seguintes parâmetros: Limite de resistência equivalente  $f_e = 1,43 \text{ N/mm}^2$ , valor da deformação equivalente a partir do qual o processo de dano se inicia  $\kappa_0 = 0,000215$  e módulo de elasticidade equivalente  $\tilde{E} = 13463,0 \text{ N/mm}^2$ , ao passo que, para compressão:  $f_e = 16,0 \text{ N/mm}^2$ ,  $\kappa_0 = 0,0022$  e  $\tilde{E} = 13463,0 \text{ N/mm}^2$ . O painel foi modelado com elementos triangulares de três nós  $T3$  em estado plano de tensão.

Para solução do modelo, empregou-se o método de controle direto de deslocamento, incrementando-se de  $0,01 \text{ mm}$  o deslocamento vertical máximo, com tolerância para convergência de  $1 \times 10^{-3}$  e carga de referência  $q = 28,0 \text{ N/mm}$ .

Nas Figuras 9.11 e 9.12, apresentam-se, respectivamente, a deformada com fator de escala igual a 100 e o dano na direção de  $\varepsilon_1$  ao longo do domínio, para os passos 9 (19,6 N/mm), 10 (21,3 N/mm), 25 (14,3 N/mm) e 200 (3,4 N/mm).



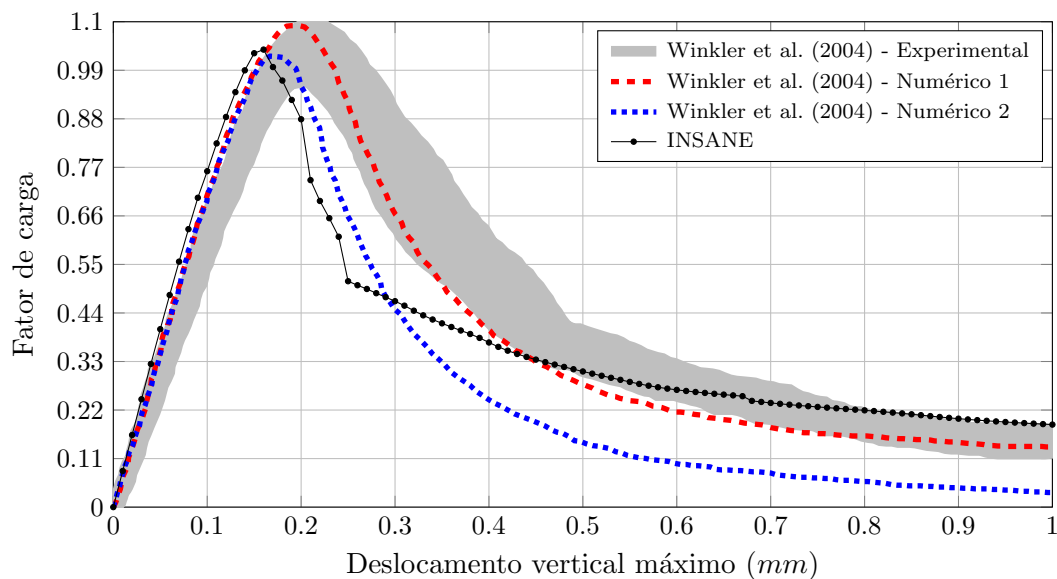
**Figura 9.11:** Painel em “L”: Deformada.



**Figura 9.12:** Pannel em “L”: Dano na direção de  $\varepsilon_1$ .

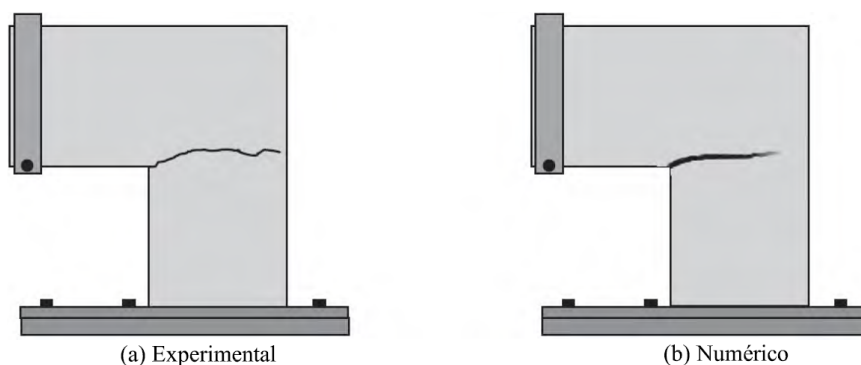


A partir das Figuras 9.11 e 9.12, destaca-se o salto no campo de deslocamentos provocado pela presença da trinca, bem como o caminhamento da trinca ao longo da região de maior danificação. Na Figura 9.13, apresenta-se a trajetória de equilíbrio do deslocamento vertical máximo, comparando-a com os resultados experimentais e numéricos de Winkler et al. (2004).



**Figura 9.13:** Painel em “L”: Trajetória de equilíbrio do deslocamento vertical máximo.

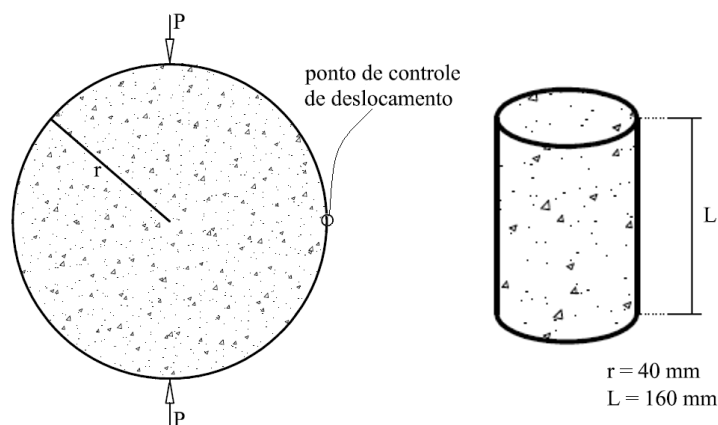
Ressalta-se que, experimentalmente, a trinca surge na junção angular do painel, o que não ocorreu nesta simulação numérica pelo fato de se assumir que o segmento da trinca no elemento inicial passa em seu centroide. Na Figura 9.14, ilustra-se o padrão de dano observado por Winkler et al. (2004).



**Figura 9.14:** Padrão de dano observado por Winkler et al. (2004).

## 9.4 Compressão Diametral

O ensaio de compressão diametral, cuja configuração é apresentada na Figura 9.15, foi simulado a partir do sistema gráfico integrado objetivando-se demonstrar o emprego deste sistema para análise de nucleação e propagação de múltiplas trincas.

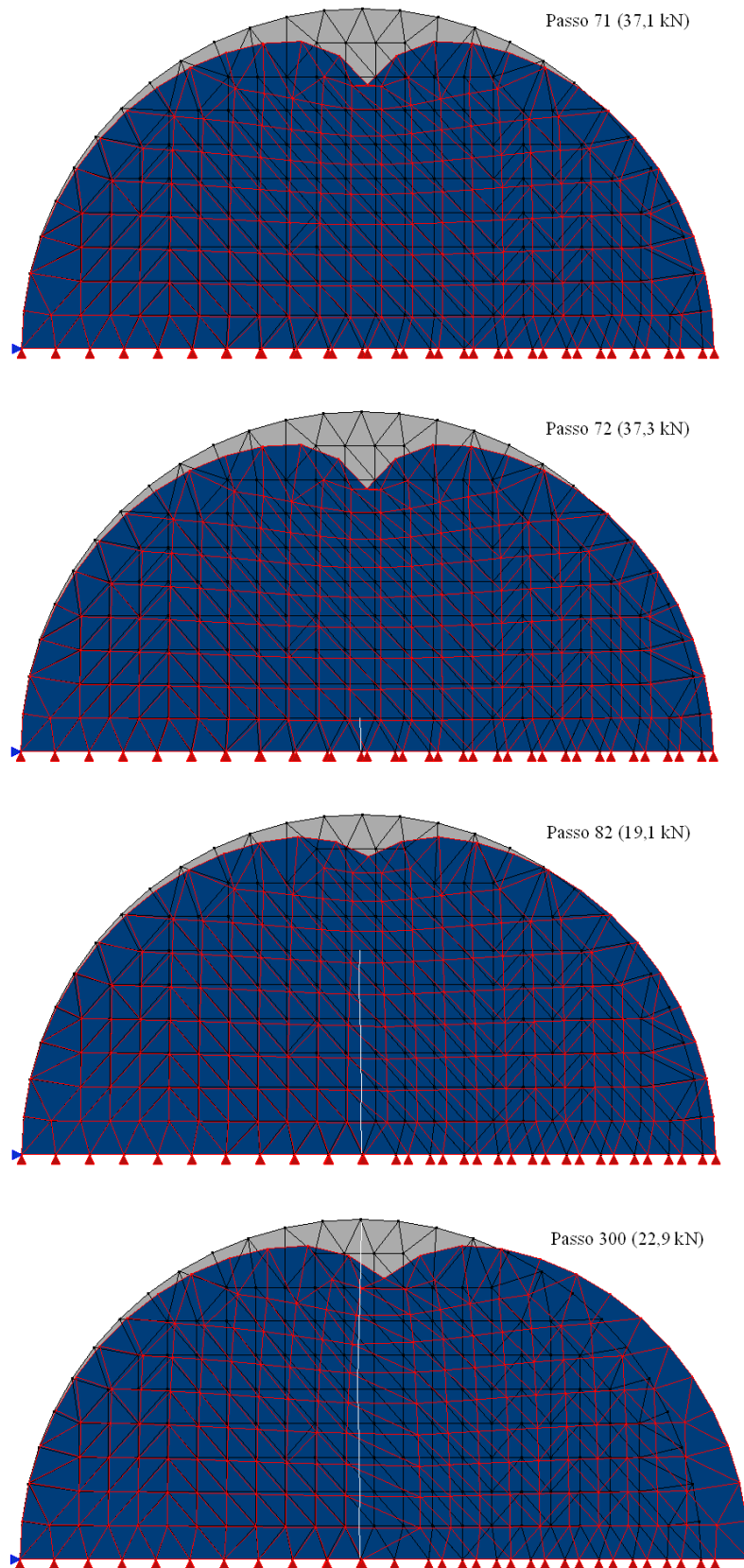


**Figura 9.15:** Compressão diametral: Configuração (Penna, 2011).

Nesta análise, adotou-se o modelo constitutivo de fissuras distribuídas com leis de Carreira e Chu (1985), para compressão, e Boone et al. (1986), para tração, cujos parâmetros do material são:  $E_0 = 20000,0 \text{ N/mm}^2$ ,  $\nu = 0,15$ ,  $f_c = 25,0 \text{ N/mm}^2$ ,  $f_t = 2,5 \text{ N/mm}^2$ ,  $\varepsilon_c = 0,01$ ,  $G_f = 0,1 \text{ N/mm}$ ,  $h = 50 \text{ mm}$  e  $\beta_r = 0,05$ . O cilindro foi modelado com elementos triangulares de três nós  $T3$  em estado plano de tensão.

Para solução do modelo, empregou-se o método de controle direto de deslocamento, incrementando-se de  $0,000075 \text{ mm}$  o deslocamento horizontal do ponto de controle destacado na Figura 9.15, com tolerância para convergência de  $1 \times 10^{-3}$  e carga de referência  $P = 0,5 \text{ N}$ .

Nas Figuras 9.16 e 9.17, apresentam-se, respectivamente, a deformada com fator de escala igual a 250 e a tensão normal horizontal  $\sigma_{xx}$  ao longo do domínio, para os passos 71 ( $37,1 \text{ kN}$ ), 72 ( $37,3 \text{ kN}$ ), 82 ( $19,1 \text{ kN}$ ) e 300 ( $22,9 \text{ kN}$ ), para o caso em que se restringiu a análise a uma única trinca, ao passo que, nas Figuras 9.18 e 9.19, apresentam-se estes campos para os passos 73 ( $37,5 \text{ kN}$ ), 82 ( $21,6 \text{ kN}$ ), 100 ( $17,9 \text{ kN}$ ) e 175 ( $20,8 \text{ kN}$ ), para o caso em que se permitiram múltiplas trincas.



**Figura 9.16:** Compressão diametral: Deformada.

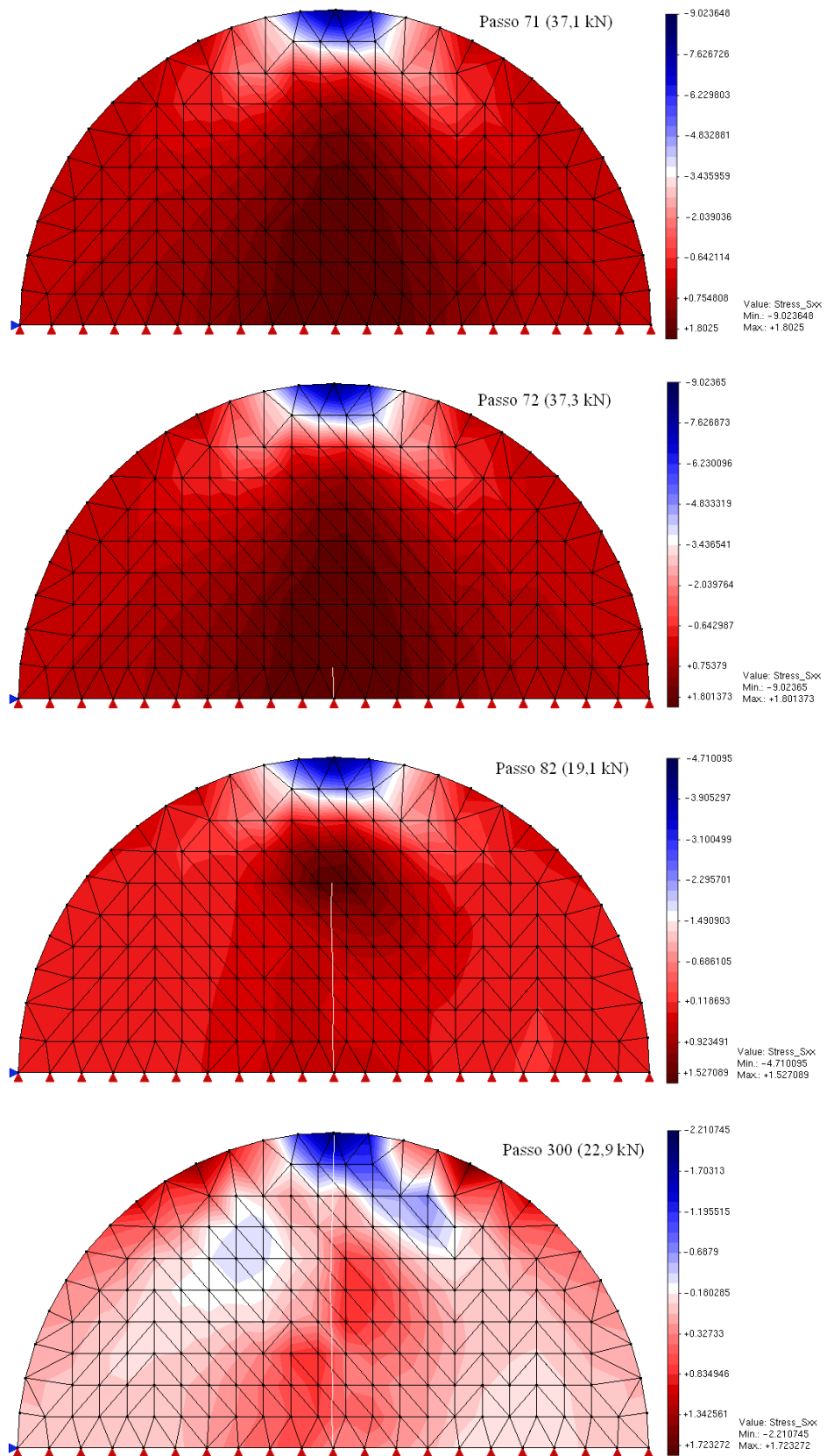
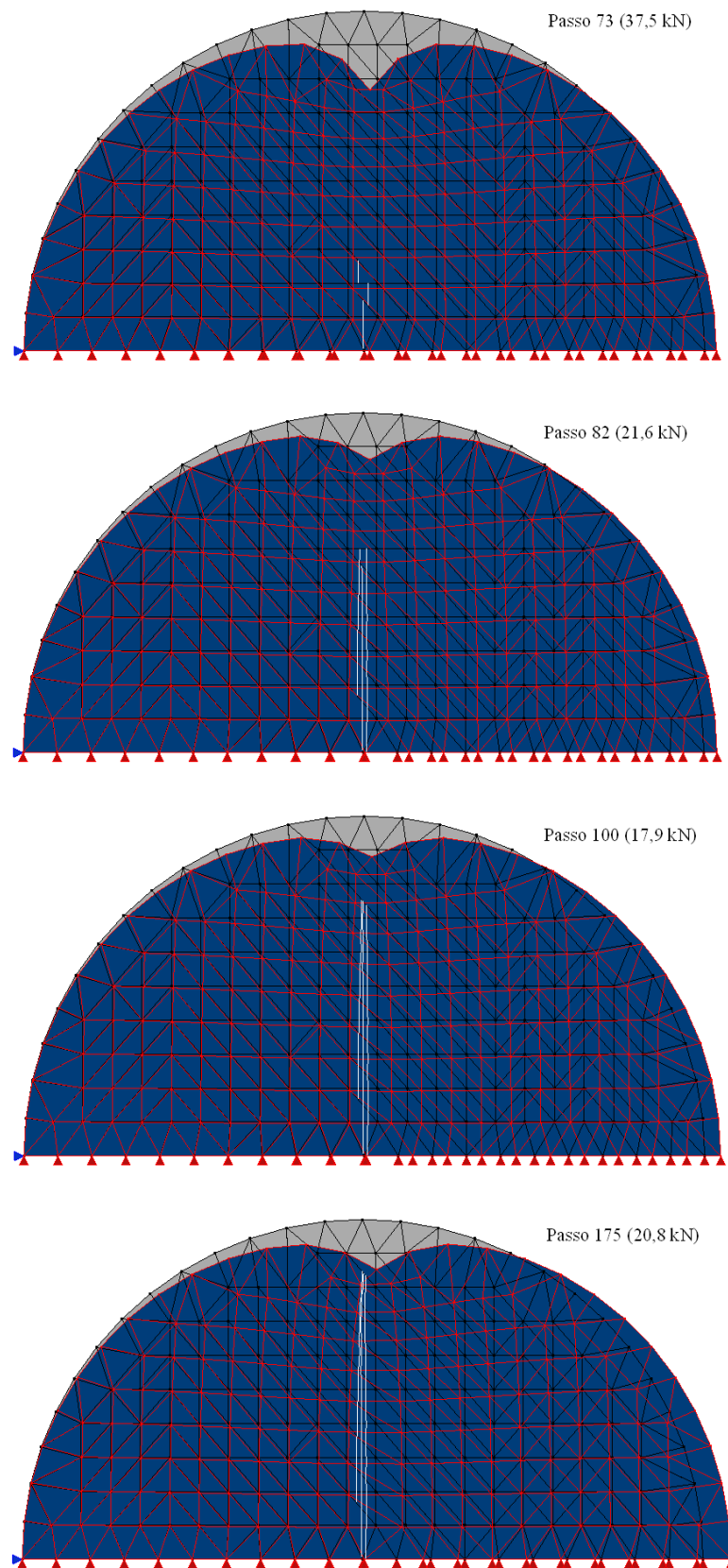


Figura 9.17: Compressão diametral: Tensão  $\sigma_{xx}$ .



**Figura 9.18:** Compressão diametral: Deformada.

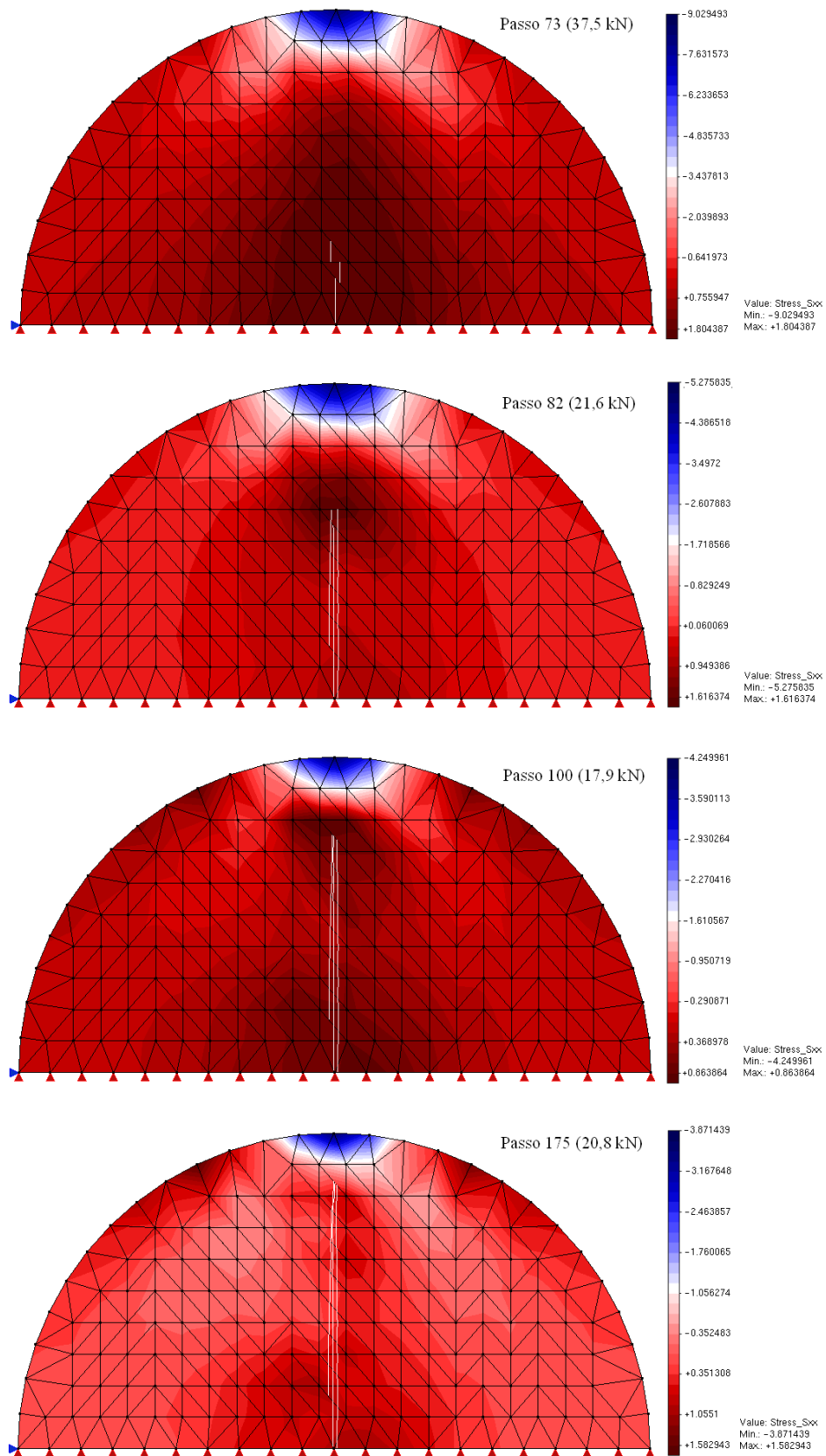
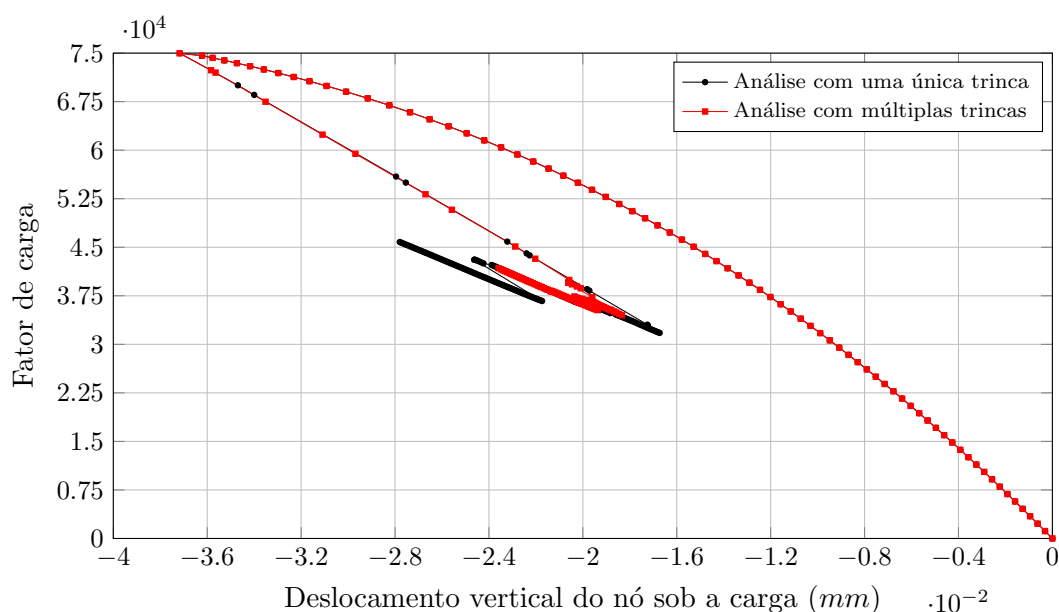


Figura 9.19: Compressão diametral: Tensão  $\sigma_{xx}$ .

É evidente que qualquer distinção entre os resultados obtidos nas análises ocorre nos passos posteriores à nucleação, ou seja, os resultados obtidos até o passo 71 (37,1 kN) são os mesmos em ambas as análises.

Porém, a partir das Figuras 9.16, 9.17, 9.18 e 9.19, destaca-se que os resultados obtidos ao longo de toda a análise com múltiplas trincas são bastante próximos daqueles obtidos na análise em que se restringiu a nucleação a uma única trinca, o que pode ser confirmado através da comparação das trajetórias de equilíbrio do deslocamento vertical do nó sob a carga, apresentada na Figura 9.20. Ressalta-se, porém, o prolongamento da análise com uma única trinca, o que as distinguem na Figura 9.20.



**Figura 9.20:** Compressão diametral: Trajetória de equilíbrio do deslocamento vertical do nó sob a carga.

Conforme em Penna (2011), a resistência da peça pode ser calculada, para o valor de carga máxima, pela equação da resistência nominal do ensaio de compressão diametral (Planas et al., 1999), dada na Equação 9.1:

$$\sigma_r = \frac{2P_{max}}{\pi D_{ef} L} \quad (9.1)$$

onde,

$\sigma_r$  é a resistência à compressão diametral;

$P_{max}$  é a carga máxima;

$D_{ef}$  é o diâmetro efetivo;

$L$  é o comprimento do corpo de prova.

Na Tabela 9.1, apresenta-se o valor da resistência à compressão diametral da peça, obtido a partir desta modelagem.

**Tabela 9.1:** Resistência à compressão diametral

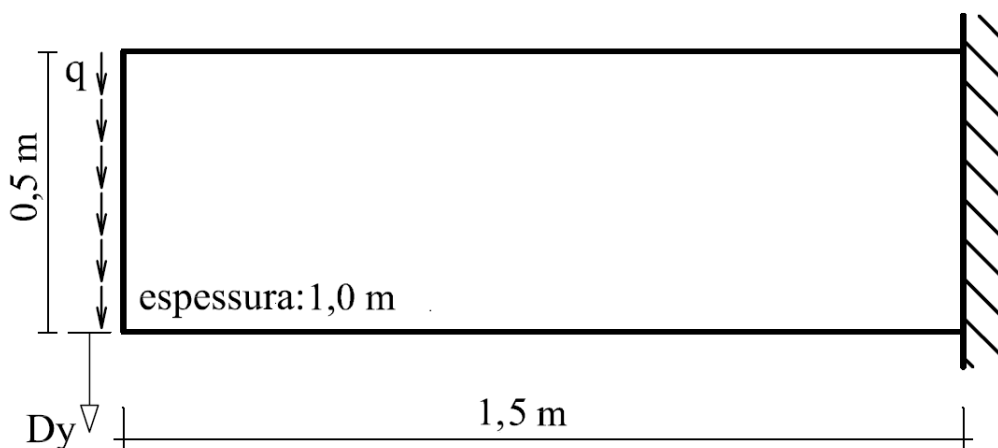
$D_{ef}$ (m)	Carga Max. (kN)	Resistência Nominal (kPa)
0,08	37,49	1864,60

Observa-se que os valores de carga máxima e resistência nominal, bem como o comportamento apresentado na Figura 9.20, são consistentes com os obtidos por Penna (2011), onde se objetivou estudar a ductilidade estrutural a partir do ensaio de compressão diametral.



## 9.5 Viga em Balanço

A viga em balanço, cuja configuração é apresentada na Figura 9.21, foi simulada a partir do sistema gráfico integrado objetivando-se demonstrar o desempenho deste sistema na análise de nucleação e propagação de trincas em elementos onde as condições de contorno essenciais são prescritas.

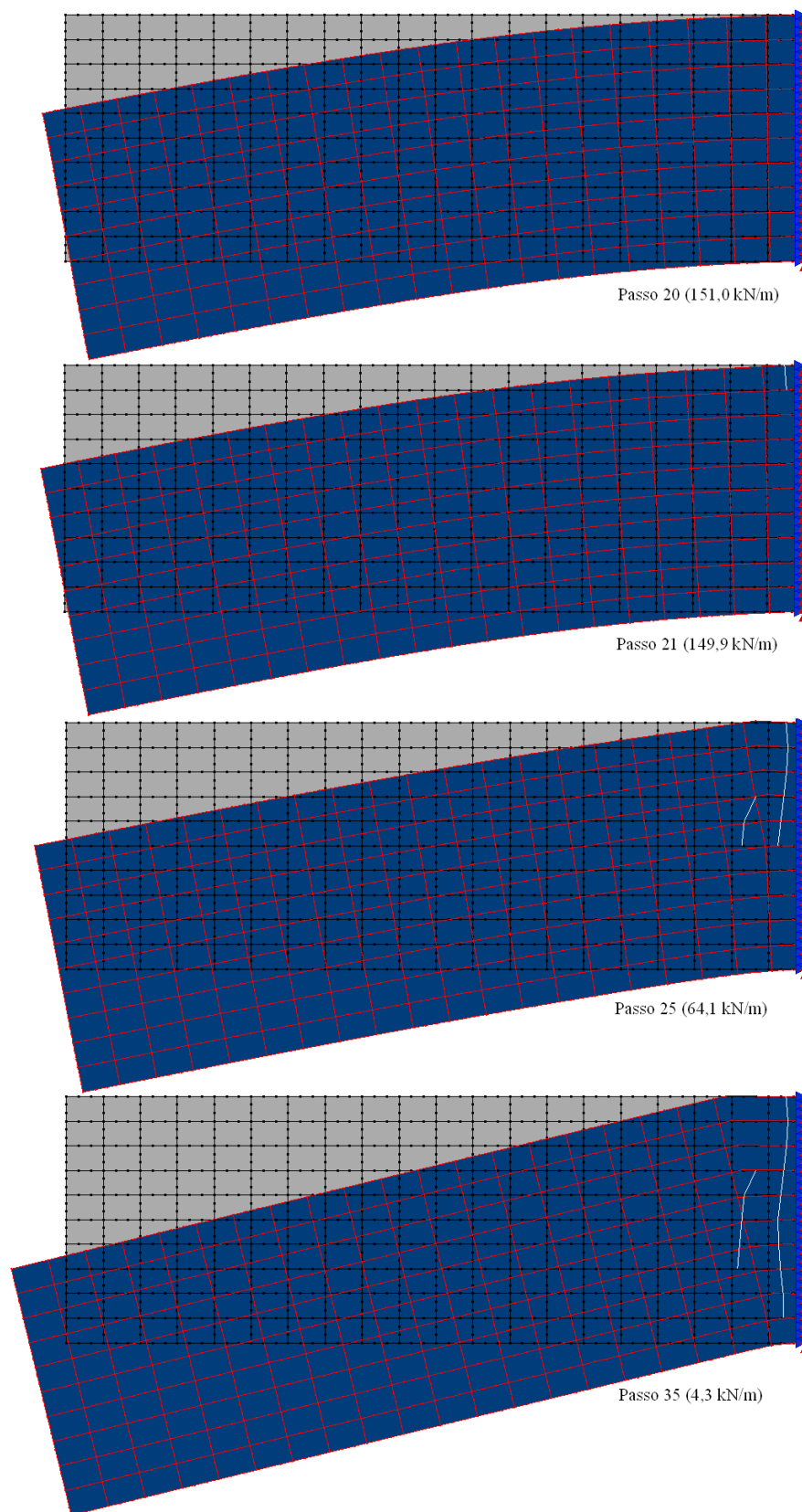


**Figura 9.21:** Viga em balanço: Configuração.

Nesta análise, adotou-se o modelo constitutivo linear elástico, cujos parâmetros do material são:  $E = 4,4 \times 10^7 \text{ kN/m}^2$ ,  $\nu = 0,2$ ,  $f_t = 3400,0 \text{ kN/m}^2$ ,  $G_f = 0,000392 \text{ kN/m}$  e  $d_{init} = 1,22 \times 10^7 \text{ kN/m}^3$ . A viga foi modelada com elementos quadrilaterais de doze nós  $Q12$  em estado plano de tensão.

Para solução do modelo, empregou-se o método de controle direto de deslocamento, incrementando-se de  $0,00001 \text{ m}$  o deslocamento vertical  $D_y$ , com tolerância para convergência de  $1 \times 10^{-4}$  e carga de referência  $q = 1,0 \text{ kN/m}$ .

Nas Figuras 9.22 e 9.23, apresentam-se, respectivamente, a deformada com fator de escala igual a 1000 e a tensão normal horizontal  $\sigma_{xx}$  ao longo do domínio, para os passos 20 ( $151,0 \text{ kN/m}$ ), 21 ( $149,9 \text{ kN/m}$ ), 25 ( $64,1 \text{ kN/m}$ ) e 35 ( $4,3 \text{ kN/m}$ ).



**Figura 9.22:** Viga em balanço: Deformada.

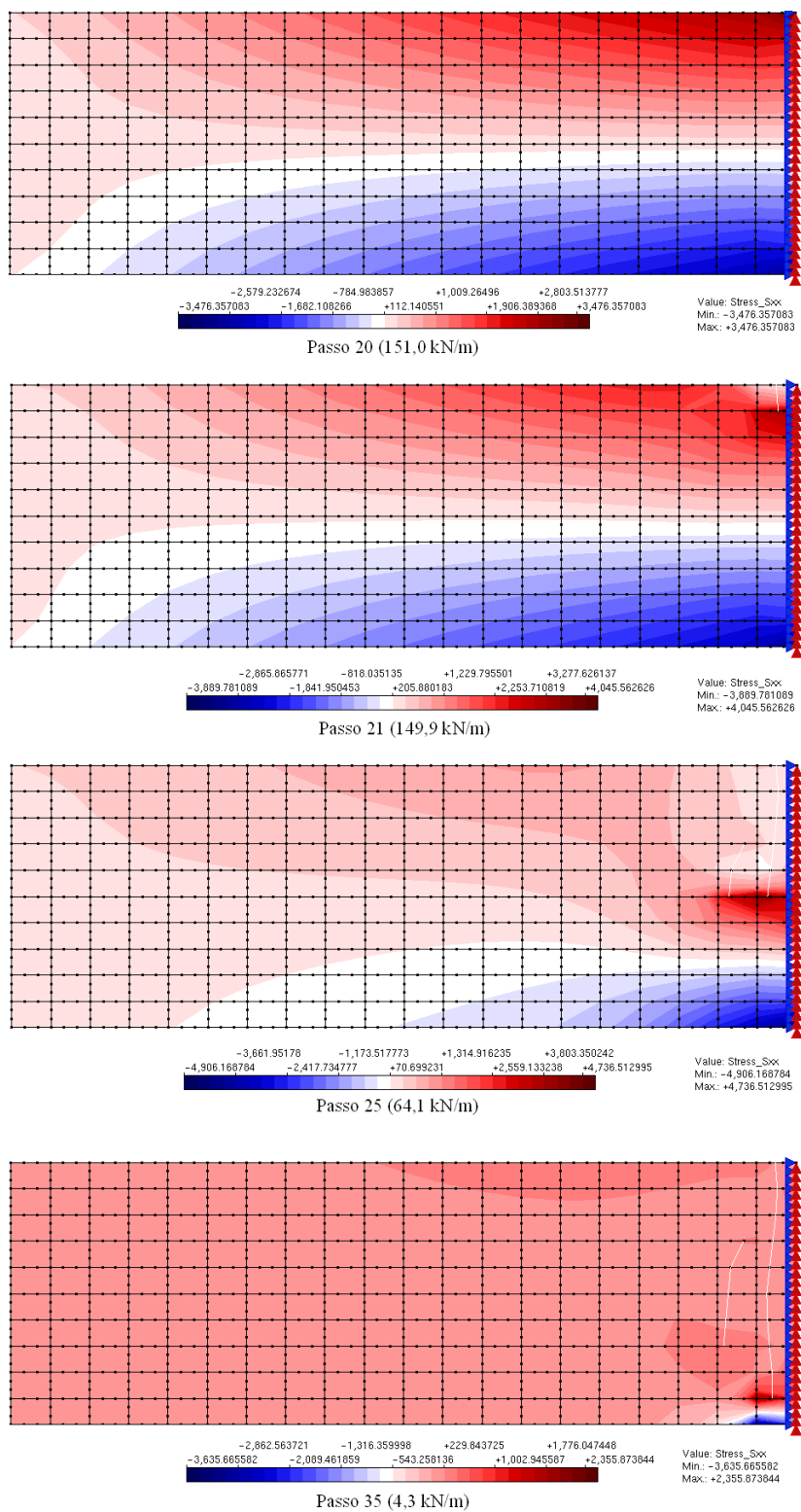
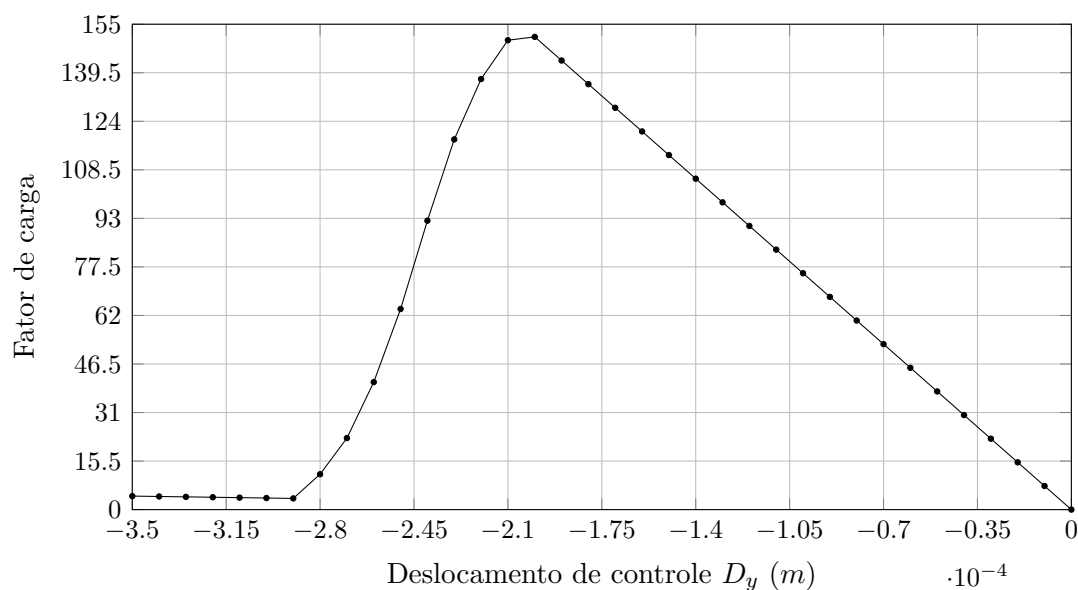


Figura 9.23: Viga em balanço: Tensão  $\sigma_{xx}$ .

A partir das Figuras 9.22 e 9.23, destaca-se a capacidade do modelo numérico aqui proposto em simular os efeitos da presença de trincas nos campos de deslocamentos, de deformações e de tensões mesmo nos casos em que alguns dos graus de liberdade adicionados para esta simulação tenham sido assumidos como zero. Na Figura 9.24, apresenta-se a trajetória de equilíbrio do deslocamento de controle  $D_y$ .

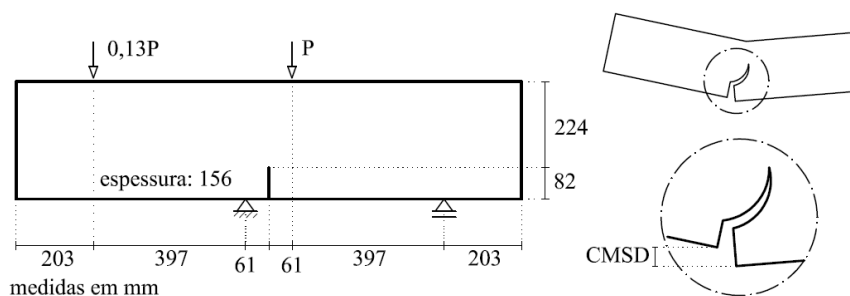


**Figura 9.24:** Viga em balanço: Trajetória de equilíbrio do deslocamento de controle  $D_y$ .

A partir das Figuras 9.23 e 9.24, observa-se que, conforme esperado para esta modelagem, a carga de pico é aquela que implica em uma máxima tensão normal na direção  $x$ ,  $\sigma_{xx}$ , aproximadamente igual à resistência à tração do material, indicando o fenômeno de nucleação.

## 9.6 Cisalhamento em 4 Pontos

A viga sob cisalhamento em 4 pontos, cuja configuração é apresentada na Figura 9.25, foi simulada a partir do sistema gráfico integrado objetivando-se demonstrar o emprego deste sistema para análise de propagação de falha preexistente.

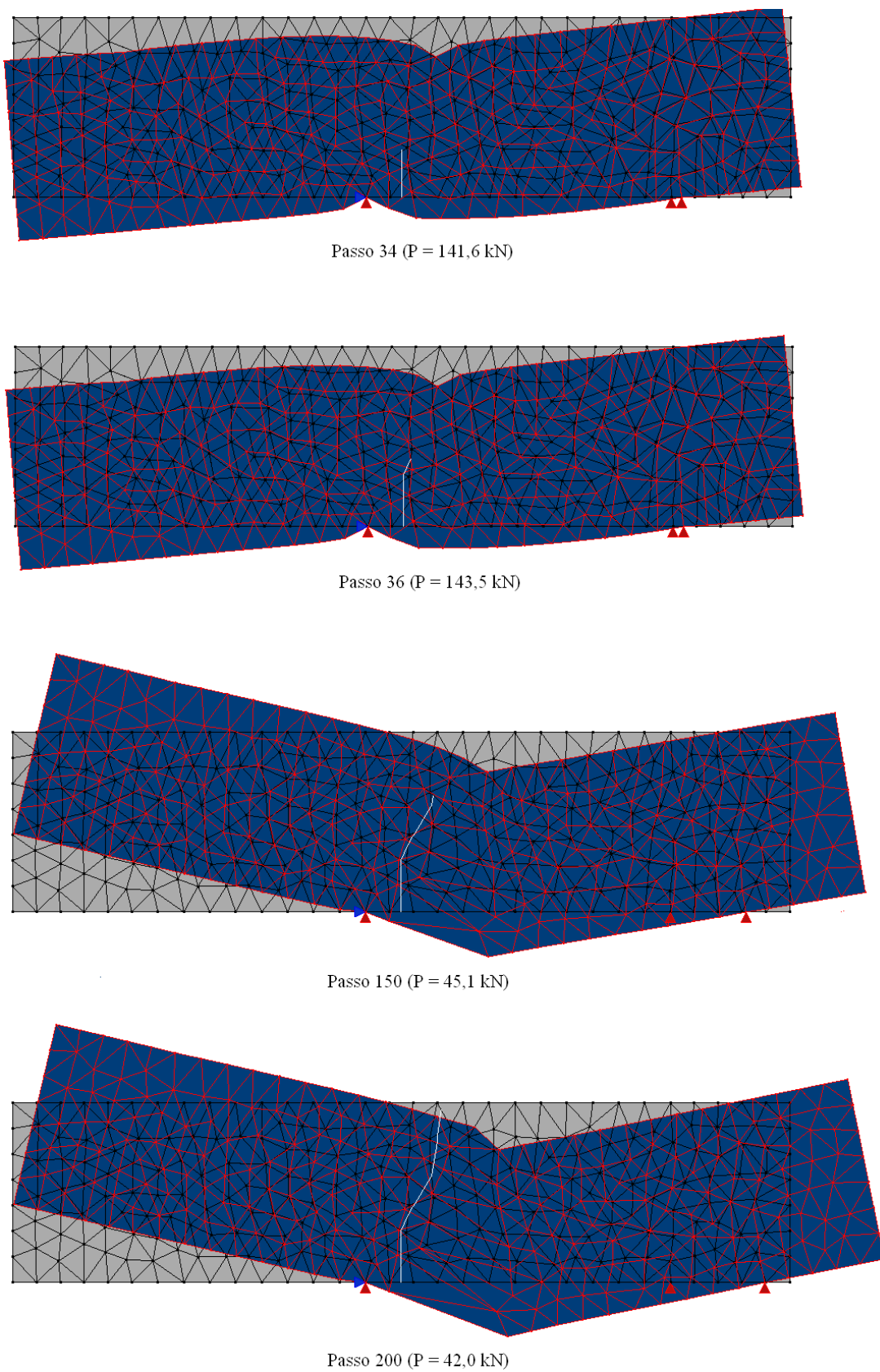


**Figura 9.25:** Cisalhamento em 4 pontos: Configuração (Penna, 2011).

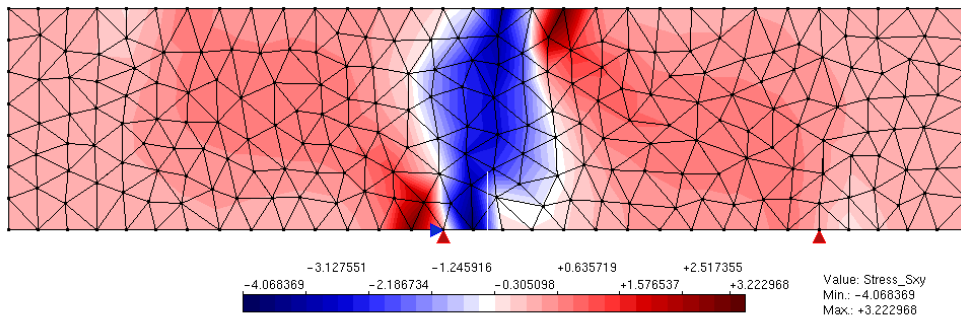
Nesta análise, adotou-se o modelo constitutivo de fissuras distribuídas com leis de Carreira e Chu (1985), para compressão, e Boone et al. (1986), para tração, cujos parâmetros do material são:  $E_0 = 24800,0 \text{ N/mm}^2$ ,  $\nu = 0,18$ ,  $f_c = 34,0 \text{ N/mm}^2$ ,  $f_t = 3,4 \text{ N/mm}^2$ ,  $\varepsilon_c = 0,002$ ,  $G_f = 0,120 \text{ N/mm}$ ,  $h = 40 \text{ mm}$  e  $\beta_r = 0,02$ . A viga foi modelada com elementos triangulares de três nós  $T3$  em estado plano de tensão.

A falha preexistente foi modelada como um entalhe a partir da definição dos seus pontos inicial (661, 0) e final (661, 82). A partir desta entrada, o programa se encarrega de enriquecer os nós necessários para simulação do salto no campo de deslocamentos. Para solução do modelo, empregou-se o método de controle direto de deslocamento, incrementando-se de  $0,00065 \text{ mm}$  o grau de liberdade adicionado a um dos nós da aresta que contém a “boca” da trinca. Este grau de liberdade refere-se a uma parcela do deslizamento tangencial relativo desta “boca”, ou seja, controlou-se um quinhão do CMSD (*Crack Mouth Sliding Displacement*), com tolerância para convergência de  $1 \times 10^{-3}$  e carga de referência  $P = 130000,0 \text{ N}$ .

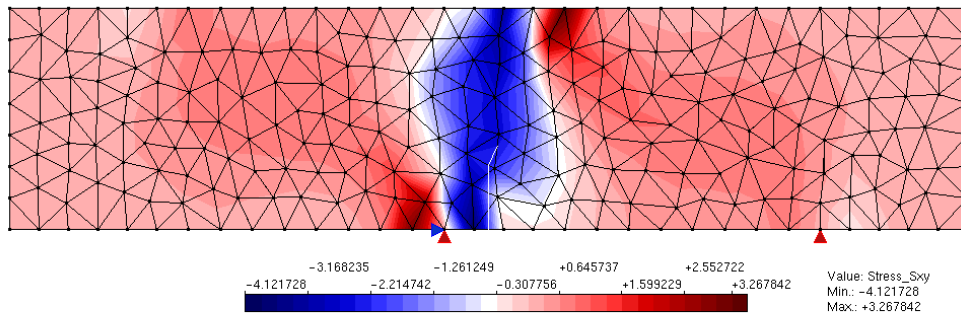
Nas Figuras 9.26 e 9.27, apresentam-se, respectivamente, a deformada com fator de escala igual a 500 e a tensão de cisalhamento  $\tau_{xy}$  ao longo do domínio, para os passos 34 (141,6 kN), 36 (143,5 kN), 150 (45,1 kN) e 200 (42,0 kN).



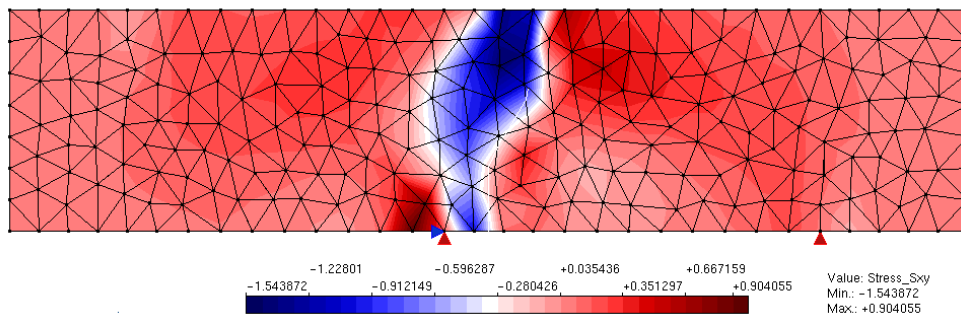
**Figura 9.26:** Cisalhamento em 4 pontos: Deformada.



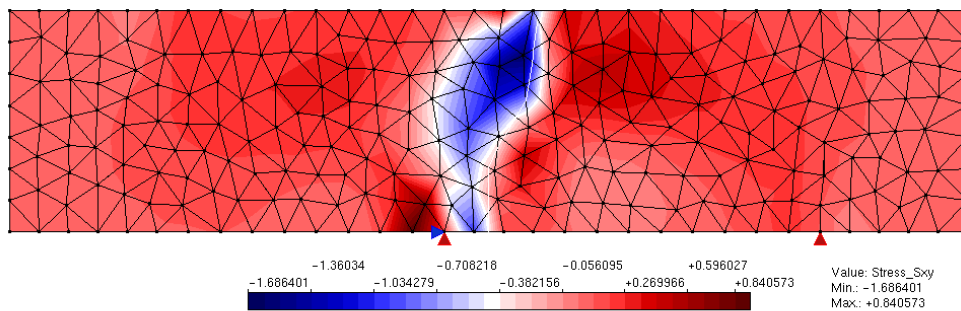
Passo 34 (P = 141,6 kN)



Passo 36 (P = 143,5 kN)



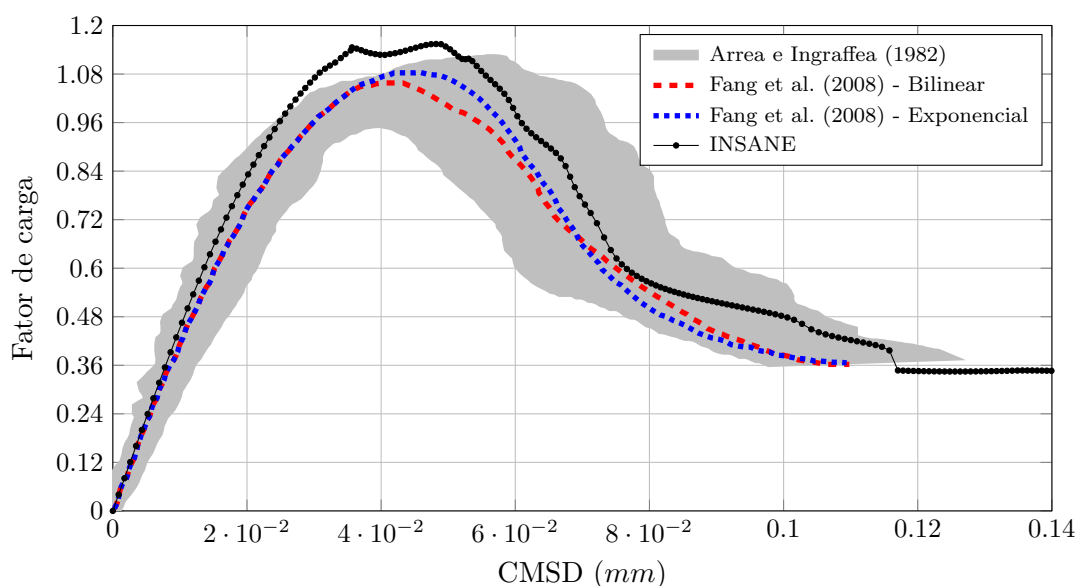
Passo 150 (P = 45,1 kN)



Passo 200 (P = 42,0 kN)

**Figura 9.27:** Cisalhamento em 4 pontos: Tensão  $\tau_{xy}$ .

A partir das Figuras 9.26 e 9.27, destaca-se a capacidade do modelo numérico aqui proposto de predição do caminho da trinca. Na Figura 9.28, apresenta-se a trajetória de equilíbrio do deslizamento tangencial relativo das extremidades da trinca (CMSD), comparando-a com os resultados experimentais de Arrea e Ingraffea (1982) e numéricos de Fang et al. (2008), que por sua vez analisaram a viga com um modelo constitutivo de trinca coesiva que adota leis exponencial e bilinear, em um modelo de elementos finitos estendidos.



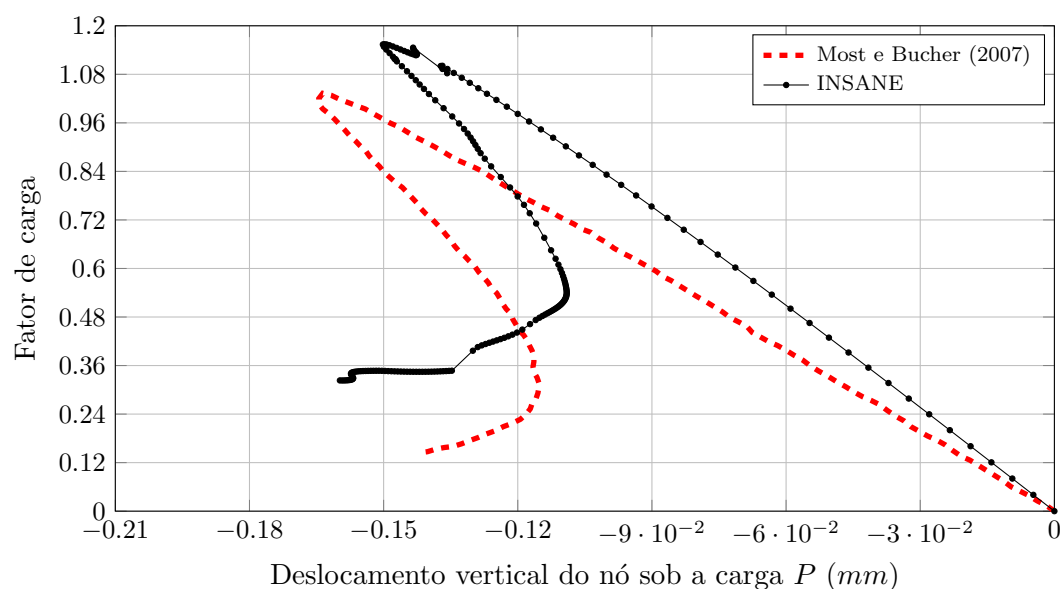
**Figura 9.28:** Cisalhamento em 4 pontos: Trajetória de equilíbrio do deslizamento tangencial relativo das extremidades da trinca (CMSD).

Observa-se que o modelo foi capaz de representar os resultados experimentais, descrevendo adequadamente a evolução do deslizamento tangencial relativo das extremidades da trinca.

Conforme Penna (2011), é relevante observar a trajetória de equilíbrio associada ao deslocamento vertical no ponto de aplicação da carga central. O experimento de Arrea e Ingraffea (1982) não apresenta resultados desta grandeza, entretanto diversas simulações desse ensaio constataram um acentuado “snap-back” para a referida trajetória.



Na Figura 9.29, apresenta-se a trajetória de equilíbrio do deslocamento vertical do nó sob a carga  $P$ , comparando-a com o resultado numérico de Most e Bucher (2007), que fizeram uso de um modelo de trinca coesiva em um método sem malha.



**Figura 9.29:** Cisalhamento em 4 pontos: Trajetória de equilíbrio do deslocamento vertical do nó sob a carga  $P$ .

A partir da Figura 9.29, pode-se verificar a equivalência dos resultados e a adequada descrição do “snap-back”, típico deste tipo de simulação, porém, observa-se que o modelo numérico proposto neste trabalho apresentou um comportamento mais rígido que o proposto por Most e Bucher (2007). Esta diferença é creditada ao fato de o modelo aqui proposto ser mais adequado para abertura de trinca em modo I predominante, ao passo que o modelo de Most e Bucher (2007) é adequado ao modo misto, de tal forma que o comportamento de amolecimento na interface é dirigido também pelo deslizamento por cisalhamento das superfícies da fissura.

# Capítulo 10

## CONSIDERAÇÕES FINAIS

### 10.1 Contribuições deste Trabalho

Neste trabalho, uma visão geral da plataforma computacional INSANE é dada com ênfase no desenvolvimento de um sistema gráfico integrado. O sistema integra todos os recursos do núcleo numérico com aplicações gráficas de pré e pós processamento. O pré-processador permite gerar uma malha de elementos finitos, com base em mapeamento transfinito ou em decomposição de domínio, a partir de um modelo geométrico bidimensional previamente definido. Além dos atributos padrões de um modelo de elementos finitos, o pré-processador permite também combinar quaisquer tipos de modelo de análise, modelo constitutivo e processo incremental-iterativo a fim de gerar um modelo linear ou fisicamente não linear. Após a definição deste modelo, o núcleo numérico do sistema INSANE é acionado para realizar a análise. Ao final do processamento, os resultados podem ser visualizados utilizando-se as diversas ferramentas do pós-processador.

Conforme destacado, o sistema gráfico integrado permite detectar nucleação de trincas e processar suas propagações. O modelo numérico implementado conjuga o modelo físico de fissura coesiva com o MEEG. Este modelo combina as características do MEF com as do MEEG, bem como dos modelos de degradação elástica com modelos de representação não geométrica baseados em funções interpoladoras com descontinuidades embutidas e estas funções de descontinuidades com leis tensão-abertura representativas das forças de coesão atuantes nos planos das trincas.

Tais combinações são monitoradas por uma medida de localização de deformações baseada na singularidade do tensor acústico. Por ser independente do modelo constitutivo e da lei de forças coesivas adotados, esta medida de nucleação resguarda a generalidade do sistema, que é caracterizada pela liberdade dada ao usuário para escolha: dos recursos para modelagem geométrica, dos métodos de geração de malhas, dos tipos de elementos finitos, dos tipos de modelo de análise, dos modelos constitutivos para acompanhamento da degradação do meio, dos processos incrementais-iterativos de análise não linear, das leis de forças coesivas atuantes nas trincas e dos recursos de visualização do pós-processador.

Os recursos implementados permitem uma aproximação com boa precisão da resposta de fissuração, sendo capaz de modelar fissuras discretas que atravessam a malha de forma independente de sua geometria e topologia. A geometria da fissura, formada por segmentos lineares, permite a propagação de fissuras curvas ou retilíneas. Além disto, resultados satisfatórios são alcançados mesmo com o uso de malhas grosseiras. Os resultados numéricos apresentados neste trabalho permitem concluir que o modelo aqui implementado apresenta similitude com os modelos experimentais. Este modelo tem como principal vantagem a não necessidade de definição prévia do caminho da trinca ou de redefinição da malha durante o processamento.

Desta forma, este trabalho contribuiu com o aperfeiçoamento da aplicação gráfica interativa de entrada de dados para análise bidimensional do sistema INSANE, bem como com o desenvolvimento de um arcabouço para análise de nucleação e propagação de trincas via abordagem de enriquecimento nodal.

## 10.2 Sugestões para Trabalhos Futuros

Com base neste trabalho, sugerem-se as seguintes expansões para o sistema IN-SANE:

1. Aperfeiçoar a aplicação gráfica interativa de entrada de dados para análise tridimensional e combinada (uni, bi e tridimensional);
2. Ampliar o arcabouço desenvolvido neste trabalho para análise de nucleação e propagação de trincas em estruturas de concreto armado;
3. Ampliar o arcabouço desenvolvido neste trabalho para análise de nucleação e propagação de trincas por fadiga;
4. Ampliar o arcabouço desenvolvido neste trabalho para análise de nucleação e propagação de trincas em meio tridimensional;
5. Avaliar a viabilidade do uso da integral  $J$  como a medida de propagação de trincas;
6. Estudar a possibilidade de se parar a propagação de trincas no interior do elemento e não, obrigatoriamente, sobre uma de suas arestas.

# Apêndice A

## Processos Incrementais-Iterativos

Numa análise não linear, confronta-se com o problema de resolver o sistema com  $N + 1$  incógnitas, sendo  $N$  deslocamentos incrementais e um incremento no fator de carga, e  $N + 1$  equações, sendo  $N$  equações de equilíbrio e uma equação de restrição. Para obter a solução deste sistema, é necessário utilizar um processo incremental-iterativo (Fuina, 2009).

Conforme Fuina (2009), o método de Newton-Raphson Padrão, um dos processos incrementais-iterativos implementados no sistema INSANE, pressupõe que a matriz de rigidez tangente é recalculada a cada iteração. Neste caso, a equação de equilíbrio incremental correspondente a iteração  $j$  do passo  $i$  pode ser escrita conforme a Equação A.1:

$$[K]_{j-1}^i \cdot \{\delta U\}_j^i = \delta \lambda_j^i \cdot \{P\} + \{Q\}_{j-1}^i \quad (\text{A.1})$$

onde,

$[K]_{j-1}^i$  é a matriz de rigidez tangente da iteração  $j - 1$  do passo  $i$ , função do campo de deslocamentos  $\{U\}_{j-1}^i$ ;

$\{\delta U\}_j^i$  é o vetor de deslocamentos incrementais da iteração  $j$  do passo  $i$ ;

$\delta \lambda_j^i$  é o incremento do fator de carga da iteração  $j$  do passo  $i$ ;

$\{P\}$  é o vetor de cargas de referência;

$\{Q\}_{j-1}^i$  é o vetor de forças residuais da iteração  $j - 1$  do passo  $i$ .

Estabelecido um valor para  $\delta\lambda_j$  em função do método de controle, pode-se obter  $\{\delta U\}_j$  conforme Equação A.2, em função das parcelas associadas à carga de referência  $\{\delta U\}_j^P$  e à carga residual  $\{\delta U\}_j^Q$ , obtidas através das Equações A.3 e A.4, respectivamente.

$$\{\delta U\}_j = \delta\lambda_j \cdot \{\delta U\}_j^P + \{\delta U\}_j^Q \quad (\text{A.2})$$

$$[K]_{j-1} \cdot \{\delta U\}_j^P = \{P\} \quad (\text{A.3})$$

$$[K]_{j-1} \cdot \{\delta U\}_j^Q = \{Q\}_{j-1} \quad (\text{A.4})$$

A atualização das variáveis fator de carga,  $\lambda_j$ , e deslocamentos,  $\{U\}_j$ , é feita através das Equações A.5 e A.6, respectivamente.

$$\lambda_j = \lambda_{j-1} + \delta\lambda_j \quad (\text{A.5})$$

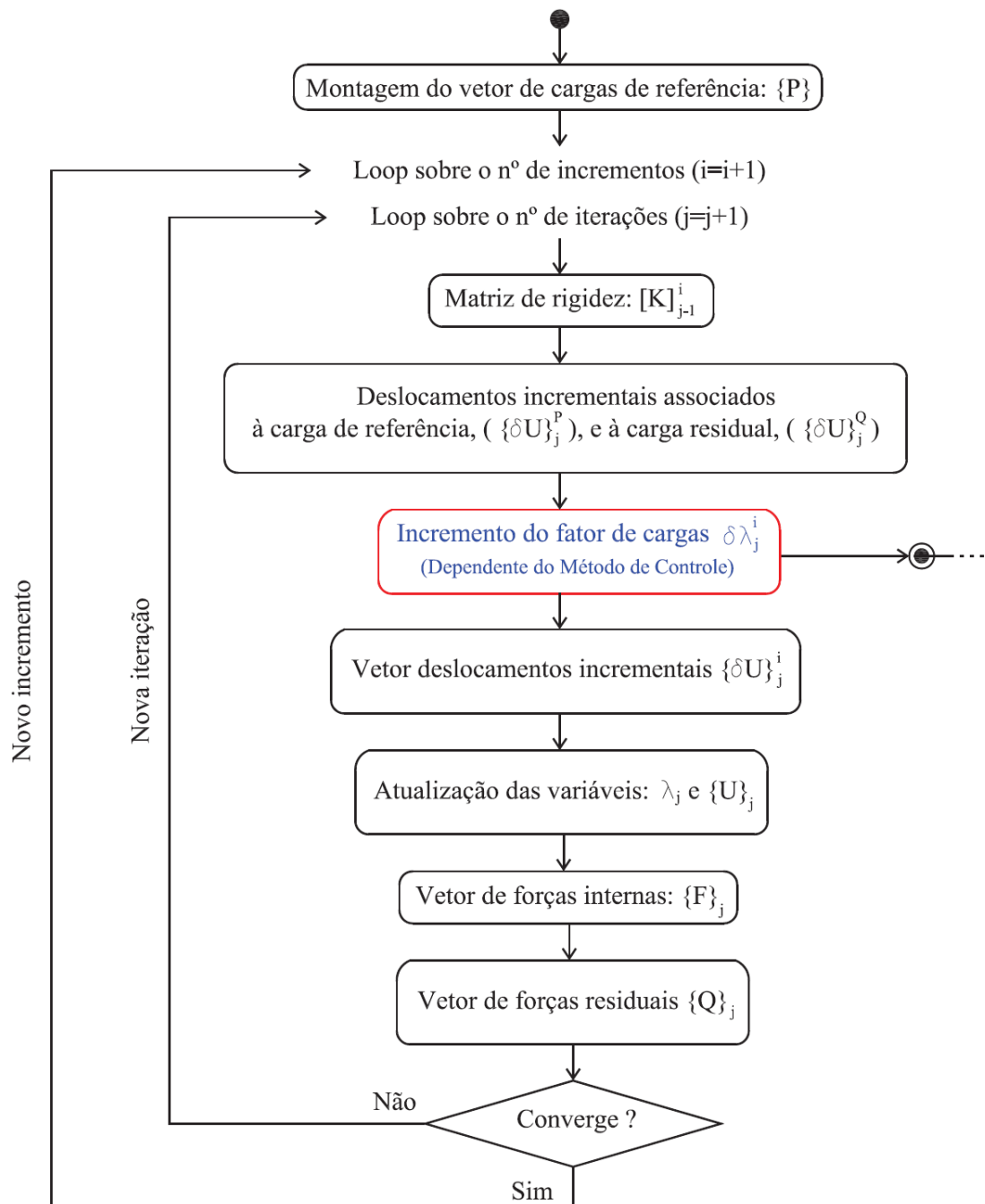
$$\{U\}_j = \{U\}_{j-1} + \{\delta U\}_j \quad (\text{A.6})$$

O vetor de cargas residuais,  $\{Q\}_j$ , é obtido a partir do vetor de forças equivalentes às tensões internas ao final da iteração  $j$ ,  $\{F\}_j$ , conforme Equação A.7, exceto na primeira iteração de cada passo, quando é assumido nulo.

$$\{Q\}_j = \lambda_j \cdot \{P\} - \{F\}_j \quad (\text{A.7})$$

Ao final de cada iteração, a convergência é verificada por meio da magnitude do vetor de forças residuais  $\{Q\}_j$  e/ou da magnitude do vetor de deslocamentos incrementais  $\{\delta U\}_j$ . O processo iterativo continua até que a convergência seja alcançada.

O diagrama da Figura A.1 mostra os principais passos do algoritmo genérico proposto por Yang e Shieh (1990), relativo a este processo incremental-iterativo.



**Figura A.1:** Algoritmo para métodos de controle (Fuina, 2009).

O procedimento em destaque refere-se à obtenção do incremento do fator de carga,  $\delta\lambda_j^i$ , através de uma equação de restrição que envolve combinações das grandezas do problema, definida pelo método de controle adotado.

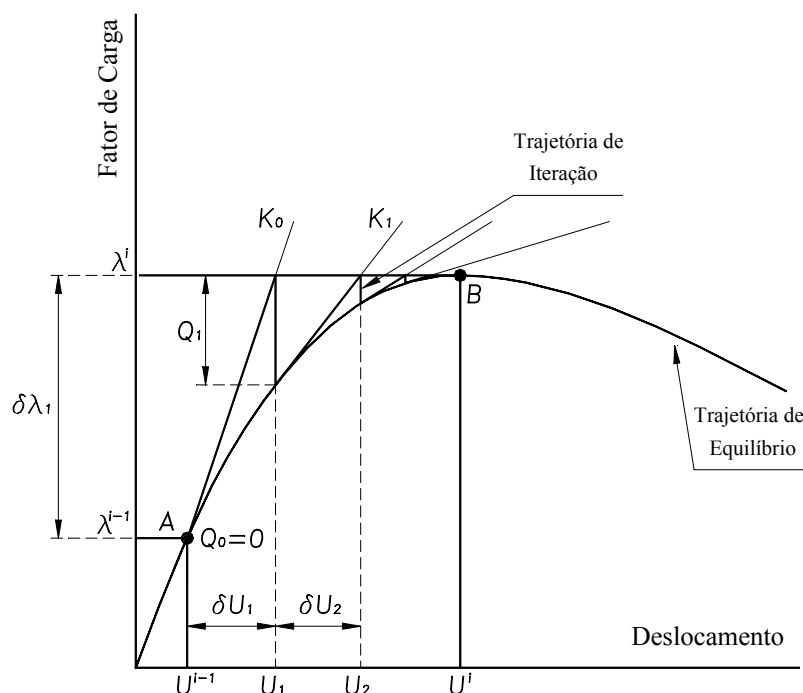
Esta formulação é bastante genérica e se aplica a vários métodos de controle, bastando que se redefina a equação de restrição. Para ilustrar o processo de equacionamento para obtenção do incremento do fator de carga,  $\delta\lambda_j^i$ , as formulações para os métodos de controle de carga e direto de deslocamento são apresentadas a seguir.

## A.1 Controle de Carga

Neste método, a carga externa é incrementada de um valor constante somente na primeira iteração de cada passo ( $j = 1$ ), sendo o incremento de carga feito igual a zero para as demais iterações ( $j > 1$ ), implicando em um carregamento externo sempre constante. A variável  $\delta\lambda_j$  pode ser obtida pela Equação A.8.

$$\delta\lambda_j = \begin{cases} \text{Constante,} & \text{para } j = 1 \\ 0, & \text{para } j > 1 \end{cases} \quad (\text{A.8})$$

A Figura A.2 apresenta um esquema do processo iterativo deste método.



**Figura A.2:** Processo incremental-iterativo com controle de carga (Fuina, 2009).



Como as iterações são processadas à carga constante, a utilização deste método falha na passagem por pontos limites de carga, ou seja, quando a carga externa ultrapassa o valor correspondente a um ponto limite (ponto B na Figura A.2), a linha horizontal que controla a trajetória de iteração não cruza a trajetória de equilíbrio, logo nenhum ponto de convergência pode ser obtido. A instabilidade numérica ocorre próximo aos pontos limites.

## A.2 Controle Direto de Deslocamento

Este método, proposto por Batoz e Dhat (1979), supõe que as iterações são processadas a um deslocamento constante. O incremento de deslocamento para o componente  $k$ ,  $\delta U_j^k$ , é dado pela Equação A.9.

$$\delta U_j^k = \begin{cases} \text{Constante,} & \text{para } j = 1 \\ 0, & \text{para } j > 1 \end{cases} \quad (\text{A.9})$$

Substituindo o vetor de deslocamentos incrementais por seu componente  $k$  na Equação A.2, tem-se a Equação A.10.

$$\delta \lambda_j = \frac{\delta U_j^k - \delta U_j^{Q^k}}{\delta U_j^{P^k}} \quad (\text{A.10})$$

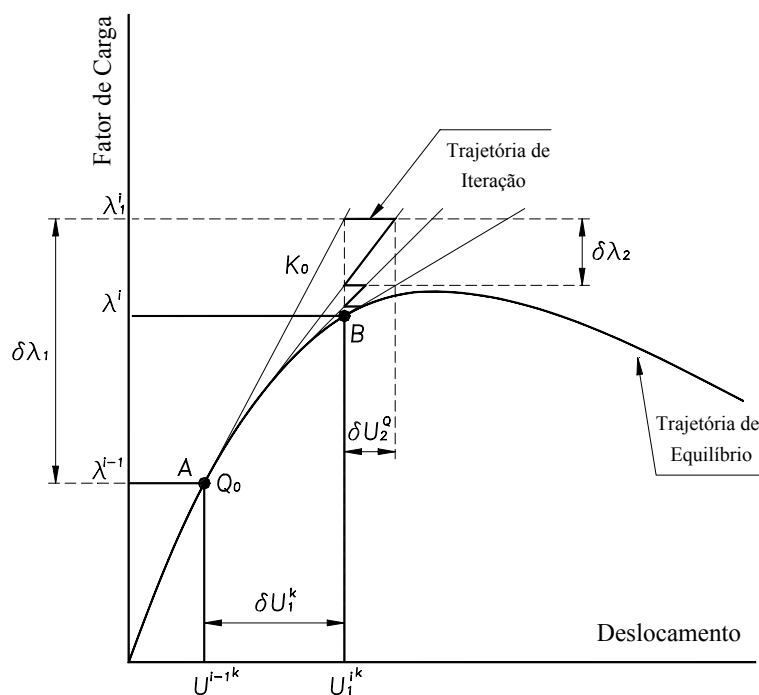
Na primeira iteração de cada passo, o vetor de cargas residuais,  $\{Q\}_{j-1}$ , é nulo, logo, os deslocamentos a ele associados,  $\{\delta U\}_j^Q$ , também são nulos, conforme a Equação A.4. Sendo assim, na primeira iteração, a Equação A.10 pode ser escrita conforme apresentado na Equação A.11.

$$\delta \lambda_1 = \frac{\delta U_1^k}{\delta U_1^{P^k}}, \quad \text{para } j = 1. \quad (\text{A.11})$$

Nas demais iterações ( $j > 1$ ),  $\delta U_j^k$  é nulo, conforme a Equação A.9, e o incremento no fator de carga pode ser escrito como na Equação A.12.

$$\delta \lambda_j = -\frac{\delta U_j^{Q^k}}{\delta U_j^{P^k}}, \quad \text{para } j > 1. \quad (\text{A.12})$$

A Figura A.3 ilustra o procedimento iterativo deste método.




**Figura A.3:** Processo incremental-iterativo com controle direto de deslocamento (Fuina, 2009).

Este método requer um conhecimento prévio da estrutura a ser analisada, para que se possa escolher o grau de liberdade adequado a ser usado para o controle. Desta forma, a experiência do usuário contribui para resolver o problema. Além disto, assim como no método de controle de carga, que não permite a passagem por pontos limites de carga, o controle direto de deslocamento é ineficiente se o deslocamento de controle experimenta diminuição (*“snap-back”*) de um nível de carga para outro. Isto se deve ao fato da trajetória de iteração, controlada por uma linha vertical, nunca cruzar a trajetória de equilíbrio.

Apesar das limitações dos métodos de controle de carga e de deslocamento, estes métodos passaram a constituir um padrão para o desenvolvimento de outros métodos mais gerais e eficazes, nos quais combinam-se deslocamentos e fator de carga.

A Figura A.4 apresenta as equações para obtenção do fator de carga nas iterações  $j = 1$  e  $j > 1$ , segundo os métodos de controle implementados no sistema INSANE.



MÉTODOS DE CONTROLE	$\delta\lambda$ para $j=1$	$\delta\lambda$ para $j>1$
Controle de Carga	$\delta\lambda_1 = \text{constante}$	$\delta\lambda_j = 0$
Controle Direto de Deslocamento	$\delta\lambda_1 = \frac{\delta U_1^k}{\delta U_1^{pk}}$	$\delta\lambda_j = -\frac{\delta U_j^{Qk}}{\delta U_j^{Pk}}$
Controle de Comprimento de Arco	$\delta\lambda_1 = \pm \frac{\Delta S}{\sqrt{\{\delta U\}_{j1}^{P^T} \cdot \{\delta U\}_{j1}^P}}$	Trajectoria de iteração ortogonal à tangente inicial: $\delta\lambda_j = -\frac{\{\Delta U\}_{j1}^T \cdot \{\delta U\}_{j1}^Q}{\{\Delta U\}_{j1}^T \cdot \{\delta U\}_{j1}^P}$
		Trajectoria de iteração ortogonal à tangente da iteração anterior: $\delta\lambda_j = -\frac{\{\Delta U\}_{j-1}^T \cdot \{\delta U\}_{j1}^Q}{\{\Delta U\}_{j-1}^T \cdot \{\delta U\}_{j1}^P}$
		Trajectoria cilíndrica: equação do 2º grau que permite obter $\delta\lambda_j$
Controle de Deslocamento Generalizado	$\delta\lambda_1 = \delta\lambda_1^1 \left( \frac{\{\delta U\}_{j1}^{P,1^T} \cdot \{\delta U\}_{j1}^{P,1}}{\{\delta U\}_{j1}^{P,i-1^T} \cdot \{\delta U\}_{j1}^{P,i}} \right)^{0,5}$	$\delta\lambda_j = -\frac{\{\delta U\}_{j1}^{P,i-1^T} \cdot \{\delta U\}_{j1}^{Q,i}}{\{\delta U\}_{j1}^{P,i-1^T} \cdot \{\delta U\}_{j1}^{P,i}}$
Controle por Trabalho	$\delta\lambda_1 = \pm \sqrt{\frac{\Delta W}{\{\delta U\}_{j1}^{P^T} \{P\}}}$	$\delta\lambda_j = -\frac{\{\delta U\}_{j1}^{Q^T} \{P\}}{\{\delta U\}_{j1}^{P^T} \{P\}}$
Método do Resíduo Ortogonal	$\delta\lambda_1 = \pm \text{constante}$	$\delta\lambda_j = \delta\lambda_1 \left( -\frac{\{\bar{Q}\}_{j1}^T \cdot \{\Delta U\}_{j1}}{\delta\lambda_1 \{P\}^T \cdot \{\Delta U\}_{j1}} \right)$

**Figura A.4:** Fatores de carga obtidos para os métodos de controle (Fuina, 2009).

No trabalho de Fuina (2009), encontram-se as formulações suprimidas deste trabalho, relativas aos equacionamentos para obtenção do incremento do fator de carga segundo os métodos de controle de comprimento de arco, com trajetória de iteração opcionalmente ortogonal à tangente inicial (Ricks, 1972, 1979), ortogonal à tangente da iteração anterior (Ramm, 1981) ou cilíndrica (Crisfield, 1981, 1983), de deslocamento generalizado (Yang e Shieh, 1990), por trabalho (Yang e McGuire, 1985) e de resíduo ortogonal (Krenk e Hededal, 1993; Krenk, 1995), para o qual são necessárias adequações no algoritmo apresentado na Figura A.1 (Fuina, 2009).

# Apêndice B

## Classe *Combinable*

No capítulo 3, apresenta-se a Metodologia de Integração do Sistema Gráfico, com enfoque na classe *Combinable*, responsável por intermediar a comunicação da aplicação gráfica de entrada de dados com o núcleo numérico do sistema. No ambiente gráfico de entrada de dados, ao se definir um parâmetro da análise e/ou determinar os seus campos, tem-se esta informação armazenada nas variáveis da classe *Combinable*. As principais variáveis da classe *Combinable* são apresentadas a seguir:

- analysisModel: modelo de análise;
- ccsUnidimensionalShape: função de forma de elementos unidimensionais em coordenadas cartesianas;
- constitutiveModel: modelo constitutivo;
- constitutiveModelLabel: rótulo do modelo constitutivo;
- degenerationsMap: tabela de armazenamento das degenerações;
- discontinuityMap: tabela de armazenamento das descontinuidades (entalhes). Estas descontinuidades são definidas por uma lista de pontos, ao passo que a simulação do salto no campo de deslocamentos é realizada através de enriquecimento nodal com função degrau, com base no MEF3D;
- enrichmentMap: tabela de armazenamento das funções de enriquecimento nodal de MEF3D;

- loadCases: lista de armazenamento dos casos de carregamento;
- loadCombinations: lista de armazenamento das combinações de carregamentos;
- loadings: lista de armazenamento dos carregamentos;
- materialsMap: tabela de armazenamento dos materiais;
- ncsQuadrilateralShape: função de forma de elementos quadrilaterais em coordenadas naturais;
- ncsTriangularShape: função de forma de elementos triangulares em coordenadas naturais;
- problemDriver: natureza do problema, cuja função é calcular as matrizes do modelo;
- shape: função de forma de elementos genéricos;
- solutionMap: tabela de armazenamento dos parâmetros da técnica de solução.

No caso de análise gerenciada pela classe *PlaneStaticAnalysis*, tem-se ainda os rótulos dos modelos constitutivos aplicáveis, além das seguintes variáveis:

- isGFEM: booleano que estabelece se a análise é pelo MEEG, definido verificando-se se a análise deve avaliar o processo de nucleação e propagação de trincas, se foram introduzidas descontinuidades (entalhes) representadas por enriquecimento nodal ou se algum nó do modelo está enriquecido;
- withCrackPropagation: booleano que estabelece se a análise deve avaliar o processo de nucleação de trincas e processar suas propagações;
- onlyOneCrack: booleano que estabelece, no caso de análise de nucleação e propagação de trincas, se somente uma trinca deve ser nucleada.

A especialização do ambiente gráfico de entrada de dados é realizada pela classe *Combinable*, responsável por filtrar do núcleo numérico, através de seus métodos, os parâmetros aplicáveis à análise que se destina. Os principais métodos da classe *Combinable* são apresentados a seguir:

- + *addDegeneration(String, String, Object)*: responsável pela criação de degenerações pelo diálogo *ModelCrossSectionListDialog*, armazenando-as na tabela *degenerationsMap*. Recebe como argumentos o nome atribuído à degeneração, o rótulo da degeneração a ser instanciada e uma tabela com os campos desta degeneração que devem ser prescritos;
- + *addDiscontinuity(String)*: responsável pela criação de descontinuidades (entalhes), simuladas pelo MEFM, pelo diálogo *SettingDiscontinuityDialog*, armazenando-as na tabela *discontinuityMap*. Recebe como argumento o nome atribuído à descontinuidade;
- + *addEnrichment(String, String)*: responsável pela criação de funções de enriquecimento nodal de MEFM pelo diálogo *SettingEnrichmentDialog*, armazenando-as na tabela *enrichmentMap*. Recebe como argumentos o nome atribuído ao enriquecimento e o rótulo da função de enriquecimento nodal a ser instanciada;
- + *addMaterial(String, String, HashMap<String, Double>)*: responsável pela criação de materiais pelo diálogo *ModelMaterialListDialog*, armazenando-os na tabela *materialsMap*. Recebe como argumentos o nome atribuído ao material, o rótulo do material a ser instanciado e uma tabela com os campos deste material que devem ser prescritos;
- + *buildFemModel(HalfEdgeDataStructure)*: responsável pela construção do objeto *Model*. Recebe como argumento a estrutura de dados para subdivisão planar, além de ter acesso às demais informações necessárias para construção deste objeto, armazenadas nas variáveis da classe;

- + `getAnalysisModel()`: retorna o modelo de análise;
- `getAnalysisModel(String)`: disparado pelo método `setAnalysisModel`, é responsável pela criação do modelo de análise em função do rótulo passado como argumento, definido no diálogo *ChooseAnalysisModelDialog*;
- + `getAnalysisModelDOFLabels()`: retorna os rótulos dos graus de liberdade nodal definidos em função do modelo de análise para especialização, por exemplo, dos diálogos de aplicação das cargas e de determinação dos atributos nodais;
- + `getAnalysisModelLabel()`: retorna o rótulo do modelo de análise;
- + `getAnalysisModelTypes()`: retorna uma lista com os rótulos dos modelos de análise aplicáveis, definidos através do método `getValidAnalysisModelsFromINSANE`. Esta lista abastece o diálogo *ChooseAnalysisModelDialog*;
- + `getCCSUnidimensionalShape()`: retorna a função de forma de elementos unidimensionais em coordenadas cartesianas;
- + `getConstitutiveModel()`: retorna o modelo constitutivo;
- + `getConstitutiveModelLabel()`: retorna o rótulo do modelo constitutivo;
- + `getConstitutiveModelTypes()`: retorna uma lista com os rótulos dos modelos constitutivos aplicáveis. Esta lista abastece o diálogo *ChooseConstitutiveModelDialog*, incluído na interface com o usuário somente se couber ao analista esta definição;
- + `getDegenerationsKeys()`: retorna uma lista com os nomes das degenerações. Esta lista abastece o painel de apresentação das degenerações incluídas no modelo, componente do diálogo *ModelCrossSectionListDialog*;
- + `getDegenerationsMap()`: retorna a tabela de armazenamento das degenerações;



- + *getDegenerationValues(String)*: retorna a tabela de armazenamento dos campos prescritíveis, bem como de seus valores, da degeneração cujo nome é passado como argumento. Esta tabela abastece o painel de entrada das propriedades da degeneração, componente do diálogo *ModelCrossSectionListDialog*;
- + *getDiscontinuitiesKeys()*: retorna uma lista com os nomes das descontinuidades (entalhes). Esta lista abastece o painel de apresentação das descontinuidades incluídas no modelo, componente do diálogo *SettingDiscontinuityDialog*;
- + *getDiscontinuityMap()*: retorna a tabela de armazenamento das descontinuidades (entalhes);
- + *getEnrichmentMap()*: retorna a tabela de armazenamento das funções de enriquecimento nodal;
- + *getEnrichmentsKeys()*: retorna uma lista com os nomes das funções de enriquecimento nodal. Esta lista abastece o painel de apresentação dos enriquecimentos incluídos no modelo, componente do diálogo *SettingEnrichmentDialog*;
- + *getInelasticLawParametersDefault(String)*: retorna uma tabela com os campos da lei inelástica que devem ser prescritos. Recebe como argumento o rótulo desta lei. Esta tabela abastece o painel de entrada dos parâmetros da lei inelástica, componente do diálogo *ModelMaterialListDialog*, se aplicável;
- + *getInstance(String)*: fábrica da classe *Combinable* cujo argumento é o rótulo da classe a ser instanciada;
- + *getInstance(String, String)*: fábrica da classe *Combinable* cujos argumentos são os rótulos da classe a ser instanciada e do modelo de análise;
- + *getInstanciables()*: retorna uma lista com os rótulos dos *Combinables* aplicáveis à análise unidimensional. Esta lista abastece o diálogo *ChooseCombinableDialog*;

- + getInstanciáveisPlane(): retorna uma lista com os rótulos dos *Combinables* aplicáveis à análise bidimensional. Esta lista abastece o diálogo *ChooseCombinableDialog*;
- + getIterativeStrategy(String): responsável pela criação do método de controle em função do rótulo passado como argumento, definido no diálogo *ProcessorDialog*;
- + getIterativeStrategyKeys(String): retorna uma lista com os campos do método de controle que devem ser prescritos. Recebe como argumento o rótulo deste método. Esta lista abastece o painel de entrada dos parâmetros do método de controle, componente do diálogo *ProcessorDialog*, se aplicável;
- + getLabel(): retorna o rótulo do *Combinable*;
- + getLoadCase(String): retorna o caso de carregamento cujo nome é passado como argumento;
- + getLoadCases(): retorna a lista de armazenamento dos casos de carregamento;
- + getLoadCasesLabels(): retorna uma lista com os nomes dos casos de carregamento. Esta lista abastece o painel de apresentação dos casos de carregamento incluídos no modelo, componente do diálogo *LoadCaseIncDialog*;
- + getLoadCombination(String): retorna a combinação de carregamentos cujo nome é passado como argumento;
- + getLoadCombinations(): retorna a lista de armazenamento das combinações de carregamentos;
- + getLoadCombinationsLabels(): retorna uma lista com os nomes das combinações de carregamentos. Esta lista abastece o painel de apresentação das combinações de carregamentos incluídas no modelo, componente do diálogo *LoadCombinationsDialog*;

- + `getLoading(String)`: retorna o carregamento cujo nome é passado como argumento;
- + `getLoadings()`: retorna a lista de armazenamento dos carregamentos;
- + `getLoadingsLabels()`: retorna uma lista com os nomes dos carregamentos. Esta lista abastece, por exemplo, o painel de apresentação dos carregamentos incluídos no modelo, componente do diálogo *LoadCaseIncDialog*;
- + `getMaterialKey(Material)`: retorna o nome do material passado como argumento;
- + `getMaterialOfDegeneration(String)`: retorna o nome do material associado à degeneração cujo nome é passado como argumento;
- + `getMaterialsKeys()`: retorna uma lista com os nomes dos materiais. Esta lista abastece, por exemplo, o painel de apresentação dos materiais incluídos no modelo, componente do diálogo *ModelMaterialListDialog*;
- + `getMaterialsMap()`: retorna a tabela de armazenamento dos materiais;
- + `getMaterialValues(String)`: retorna a tabela de armazenamento dos campos prescritíveis, bem como de seus valores, do material cujo nome é passado como argumento. Esta tabela abastece o painel de entrada das propriedades do material, componente do diálogo *ModelMaterialListDialog*;
- + `getNCSQuadrilateralShape()`: retorna a função de forma de elementos quadrilaterais em coordenadas naturais;
- + `getNCSTriangularShape()`: retorna a função de forma de elementos triangulares em coordenadas naturais;
- + `getProblemDriver()`: retorna a natureza do problema;

- + `getRotationMatrix(IMatrix)`: retorna a matriz de transformação do sistema global para o sistema local do elemento. Recebe como argumento a matriz de coordenadas nodais deste elemento. Esta matriz é empregada na composição de objetos de desenho;
- + `getShape()`: retorna a função de forma de elementos genéricos;
- + `getSolution()`: responsável pela construção do objeto *Solution* a partir das entradas no diálogo *ProcessorDialog*;
- + `getSolutionMap()`: retorna a tabela de armazenamento dos parâmetros da técnica de solução;
- + `getSolutionName()`: retorna o rótulo da técnica de solução. Este rótulo é apresentado no diálogo *ProcessorDialog*;
- + `getSolutionValues()`: retorna uma lista com os campos da técnica de solução que devem ser prescritos. Esta lista abastece o painel de entrada dos parâmetros da técnica de solução, componente do diálogo *ProcessorDialog*, se aplicável;
- # `getStep(String)`: responsável pela criação do processo incremental-iterativo em função do rótulo passado como argumento, definido no diálogo *ProcessorDialog*;
- + `getUsedMaterialsList()`: retorna uma lista com os nomes dos materiais associados a alguma degeneração. Esta lista é verificada para permitir ou não a exclusão de um material pelo diálogo *ModelMaterialListDialog*;
- # `getValidAnalysisModelsFromINSANE()`: retorna uma lista com os tipos de modelo de análise aplicáveis;
- + `getValidDegenerationMap(String)`: retorna uma tabela com os campos da degeneração que devem ser prescritos. Recebe como argumento o rótulo desta degeneração;

- # *getValidDegenerationsFromINSANE()*: retorna uma lista com os tipos de degeneração aplicáveis, avaliado em função do modelo de análise;
- + *getValidDegenerationTypes()*: retorna uma lista com os rótulos das degenerações aplicáveis. Esta lista abastece a caixa de escolha do tipo de degeneração, componente do diálogo *ModelCrossSectionListDialog*;
- + *getValidInelasticLawTypes()*: retorna uma lista com os rótulos das leis inelásticas aplicáveis. Esta lista abastece a caixa de escolha do tipo de lei inelástica, componente do diálogo *ModelMaterialListDialog*, se aplicável;
- # *getValidIterativeStrategyFromINSANE()*: retorna uma lista com os tipos de método de controle aplicáveis;
- + *getValidIterativeStrategyTypes()*: retorna uma lista com os rótulos dos métodos de controle aplicáveis. Esta lista abastece a caixa de escolha do tipo de método de controle, componente do diálogo *ProcessorDialog*, se aplicável;
- + *getValidMaterialMap(String)*: retorna uma tabela com os campos do material que devem ser prescritos. Recebe como argumento o rótulo deste material;
- # *getValidMaterialsFromINSANE()*: retorna uma lista com os tipos de material aplicáveis, avaliado em função do modelo constitutivo e do modelo de análise;
- + *getValidMaterialTypes()*: retorna uma lista com os rótulos dos materiais aplicáveis. Esta lista abastece a caixa de escolha do tipo de material, componente do diálogo *ModelMaterialListDialog*;
- # *getValidShapesFromINSANE()*: retorna uma lista com os tipos de função de forma aplicáveis. Esta lista abastece, por exemplo, o diálogo *SetsTwoDimensionalElementsDialog*, incluído na interface com o usuário de análise bidimensional;

- + `getValidStepsFromINSANE()`: retorna uma lista com os rótulos dos processos incrementais-iterativos aplicáveis. Esta lista abastece a caixa de escolha do tipo de processo incremental-iterativo, componente do diálogo *ProcessorDialog*, se aplicável;
- + `removeDiscontinuity(String)`: remove da tabela *discontinuityMap* a descontinuidade (entalhe) cujo nome é passado como argumento;
- + `removeEnrichment(String)`: remove da tabela *enrichmentMap* a função de enriquecimento nodal cujo nome é passado como argumento;
- + `removeMaterial(String)`: remove da tabela *materialsMap* o material cujo nome é passado como argumento;
- + `setAnalysisModel(String)`: disparado pelo diálogo *ChooseAnalysisModelDialog*, é responsável por disparar o método *getAnalysisModel*, repassando o rótulo recebido como argumento;
- + `setConstitutiveModel(ConstitutiveModel)`: atribui à variável *constitutiveModel* o objeto passado como argumento;
- + `setConstitutiveModelLabel(String)`: atribui à variável *constitutiveModelLabel* o rótulo passado como argumento. Nos casos em que cabe ao analista a definição do modelo constitutivo, por exemplo, nas análises gerenciadas pela classe *PlaneStaticAnalysis*, este método é responsável pela criação do modelo constitutivo em função do rótulo passado como argumento, definido no diálogo *ChooseConstitutiveModelDialog*, e pela adição de um material aplicável a este modelo constitutivo, associando-o à degeneração criada na construção do *Combinable*;
- + `setDegenerationsMap(HashMap<String, Degeneration>)`: atribui à variável *degenerationsMap* a tabela passada como argumento;

- + *setDegenerationValues(String, String, double)*: responsável por receber o valor prescrito através do diálogo *ModelCrossSectionListDialog* para o campo cujo rótulo é passado como argumento e atribuí-lo à degeneração cujo nome também é passado como argumento;
- + *setDiscontinuityMap(HashMap<String, LinkedList<Vertex> >)*: atribui à variável *discontinuityMap* a tabela passada como argumento;
- + *setEnrichmentMap(HashMap<String, EnrichmentType>)*: atribui à variável *enrichmentMap* a tabela passada como argumento;
- + *setLoadCases(ArrayList<LoadCase>)*: atribui à variável *loadCases* a lista passada como argumento;
- + *setLoadCombinations(ArrayList<LoadCombination>)*: atribui à variável *loadCombinations* a lista passada como argumento;
- + *setLoadings(ArrayList<Loading>)*: atribui à variável *loadings* a lista passada como argumento;
- + *setMaterialAtDegeneration(String, String, String)*: associa o material cujo nome é passado como argumento à degeneração cujo nome e o rótulo também são passados como argumentos;
- + *setMaterialsMap(HashMap<String, Material>)*: atribui à variável *materialsMap* a tabela passada como argumento;
- + *setMaterialValues(String, String, double)*: responsável por receber o valor prescrito através do diálogo *ModelMaterialListDialog* para o campo cujo rótulo é passado como argumento e atribuí-lo ao material cujo nome também é passado como argumento;
- + *setProblemDriver(ProblemDriver)*: atribui à variável *problemDriver* o objeto passado como argumento;

- + `setShape(Shape)`: atribui à variável *shape* o objeto passado como argumento;
- + `setShapeList(ArrayList<Shape>)`: atribui às variáveis *ncsQuadrilateralShape* e *ncsTriangularShape* os correspondentes objetos armazenados na lista passada como argumento, definidos no diálogo *SetsTwoDimensionalElementsDialog*, incluído na interface com o usuário de análise bidimensional;
- + `setSolutionMap(HashMap<String, Object>)`: atribui à variável *solutionMap* a tabela passada como argumento;
- + `verifyDataConsistency(HalfEdgeDataStructure)`: disparado pelo comando *VerifyDataConsistenceCommand*, é responsável por verificar a consistência dos dados de entrada. Este método é também disparado pelo comando *ProcessorCommand*, condicionando-se o processamento da análise a esta consistência. Recebe como argumento a estrutura de dados para subdivisão planar, além de ter acesso às demais informações necessárias para esta verificação, armazenadas nas variáveis da classe.



# Apêndice C

## Sistema Gráfico Integrado

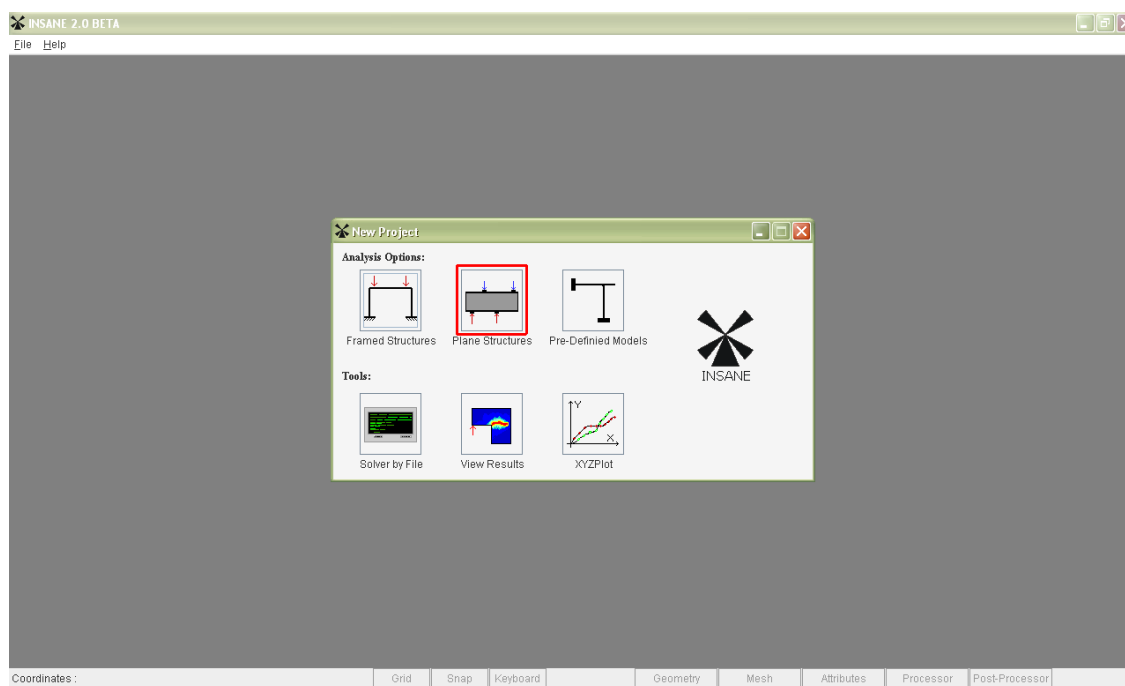
Seguindo a metodologia descrita no capítulo 3, a entrada de dados no sistema INSANE foi aperfeiçoada. Inicialmente, o ambiente gráfico de análise bidimensional foi composto por recursos e comandos para análise estática linear com carregamento no plano pelo MEF. Em seguida, este ambiente foi expandido para a opção de análise com carregamento normal ao plano (placas).

Na sequência, incluíram-se recursos e comandos para análise fisicamente não linear na opção de análise com carregamento no plano, permitindo-se que a não linearidade física seja avaliada através de modelos constitutivos que tratam a degradação do meio de forma distribuída, bem como recursos e comandos para enriquecimento nodal, permitindo-se a análise pelo MEFG.

Desta forma, permitem-se os seguintes tipos de análise bidimensional estática através do sistema gráfico integrado:

- Linear com carregamento no plano pelo MEF;
- Linear com carregamento no plano pelo MEFG;
- Fisicamente não linear com carregamento no plano pelo MEF;
- Fisicamente não linear com carregamento no plano pelo MEFG;
- Linear com carregamento normal ao plano (placas) pelo MEF.

A entrada no ambiente gráfico é realizada no diálogo inicial apresentado na Figura C.1, definindo-se a classificação espacial da análise. Conforme capítulo 3, este diálogo apresenta os recursos de uso do sistema. No estágio atual, tem-se as opções para análise unidimensional, bidimensional e de modelos pré-definidos via sistema gráfico integrado, a opção para análise unidimensional, bidimensional e tridimensional via arquivo XML e as opções para visualização gráfica dos resultados.



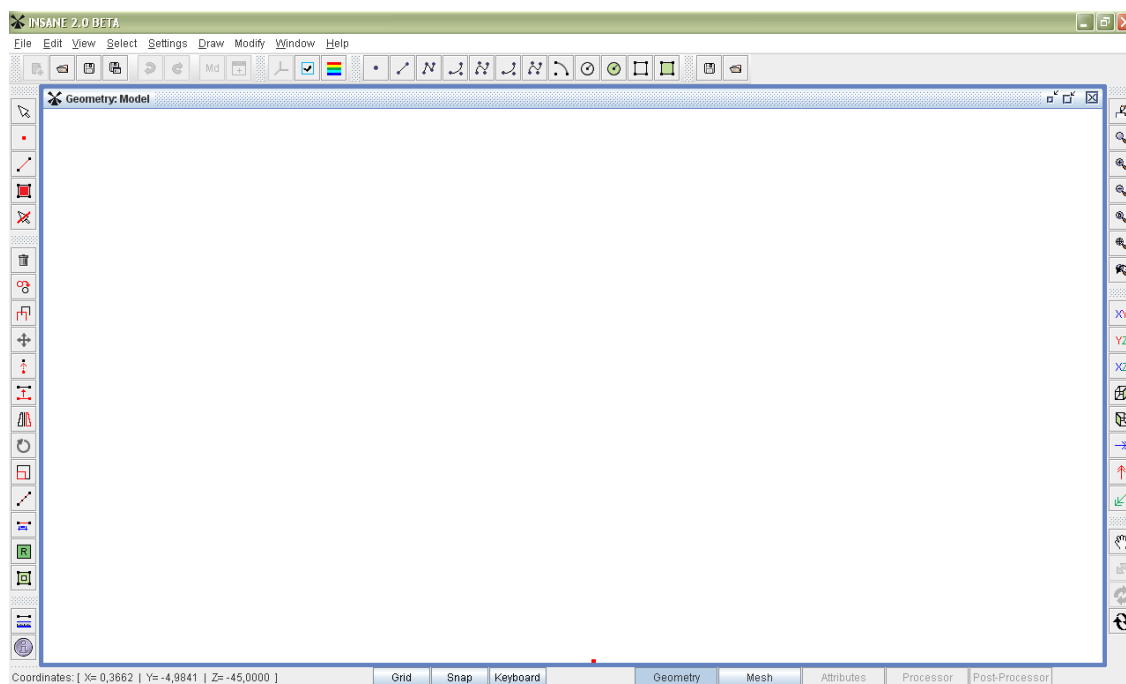
**Figura C.1:** Diálogo inicial do sistema INSANE.

Na Figura C.1, destaca-se a entrada no ambiente gráfico para a opção de análise bidimensional. Após a definição da classificação espacial da análise, define-se o diretório para armazenamento dos dados do modelo. Na sequência, abre-se automaticamente o primeiro módulo componente do pré-processador gráfico.

O pré-processador gráfico foi dividido em três módulos, a saber: Módulo *Geometria*, Módulo *Malha* e Módulo *Atributos*. Esta modularização permite, por exemplo, a definição de várias malhas diferentes para um mesmo modelo geométrico, bem como analisar um modelo já discretizado com diferentes elementos finitos. Estes módulos são apresentados a seguir, segundo a organização do programa.

## C.1 Módulo *Geometria*

O Módulo *Geometria*, apresentado na Figura C.2, é o módulo inicial do pré-processador gráfico. É composto de recursos para desenho, seleção e visualização de entidades geométricas. Conforme definição, neste módulo, desenha-se a geometria da estrutura em análise.



**Figura C.2:** Pré-processador gráfico interativo: Módulo *Geometria*.

A barra de ferramentas superior disponibiliza recursos para geração de primitivas geométricas, tais como: pontos, linhas, arcos, círculos, curvas cúbicas, curvas quadráticas e regiões retangulares, circulares e poligonais.

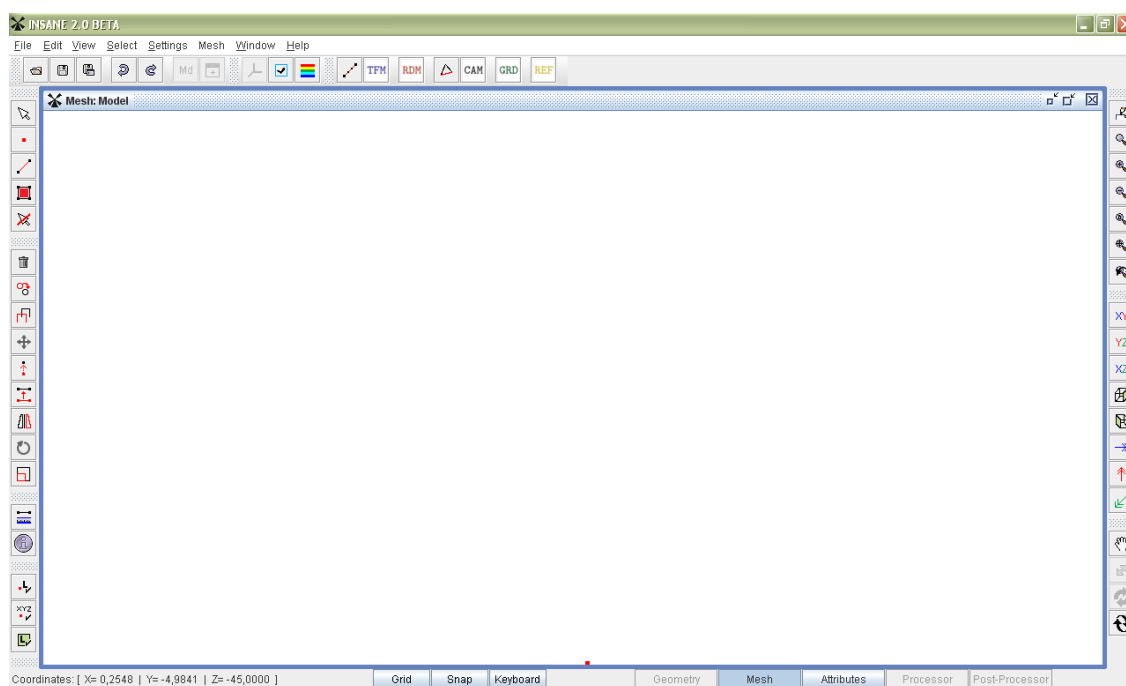
A barra de ferramentas lateral esquerda disponibiliza recursos para seleção e edição de entidades geométricas, ao passo que a barra de ferramentas lateral direita disponibiliza recursos de visualização.

A barra de status, disposta na parte inferior da tela, é responsável por gerenciar o fluxo de utilização no ambiente gráfico. Conforme Figura C.2, esta barra destaca o módulo corrente (*Geometria*), bem como o botão de fluxo acionável (*Malha*) que encapsula o comando responsável por disparar a passagem para o próximo módulo.

Com a geometria definida, segue-se para o Módulo *Malha*. Este fluxo pode ocorrer para o número de malhas que o analista quiser processar, cumulando janelas deste módulo cujas estruturas de dados associadas, que serão modificadas quando da geração da malha, são independentes entre si, devido à clonagem da estrutura de dados na mudança de módulos.

## C.2 Módulo *Malha*

O Módulo *Malha*, apresentado na Figura C.3, é composto, além dos recursos para seleção e visualização, de recursos para geração de malhas. Conforme definição, neste módulo, discretiza-se a geometria da estrutura em análise. No estágio atual do sistema, são permitidas as seguintes técnicas de geração de malhas: Mapeamento Transfinito (Fonseca, 1989), Geração Nodal Aleatória (Ho-Le, 1988), Malha Assimétrica Concentrada e Malha com base em Grade (Ho-Le, 1988). Conforme seção 2.4, o primeiro é uma técnica de mapeamento da fronteira, enquanto os demais são técnicas de decomposição de domínio.

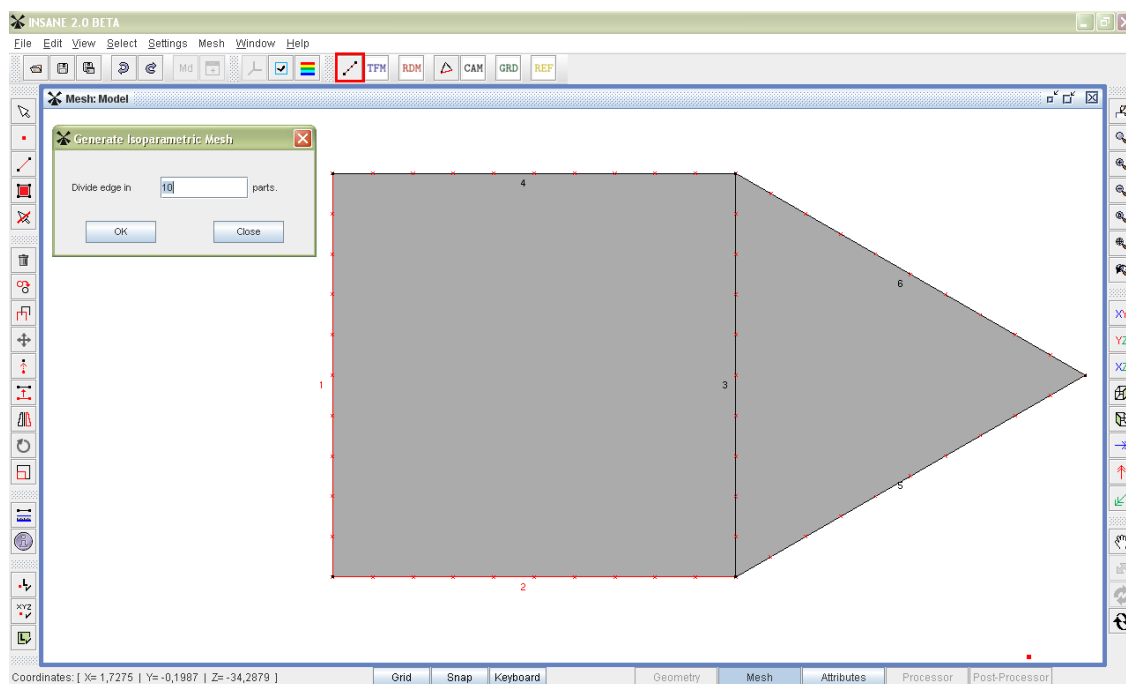


**Figura C.3:** Pré-processador gráfico interativo: Módulo *Malha*.

A barra de ferramentas superior disponibiliza os recursos para geração de malhas, cujas funcionalidades são apresentadas a seguir.

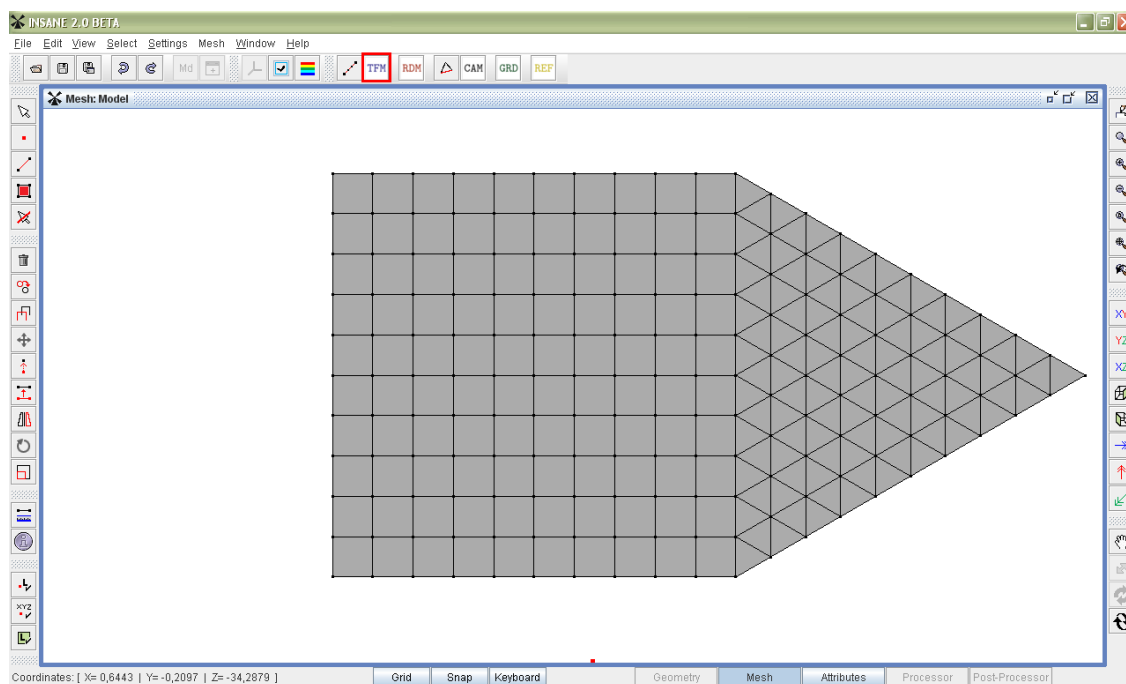
No caso de Mapeamento Transfinito, o processo de discretização é realizado definindo-se o número de divisões de determinadas arestas. Visando atender ao critério da continuidade inerente ao método discreto, o programa se encarrega da propagação das divisões necessárias para as demais arestas do modelo geométrico.

Este processo de discretização é ilustrado nas Figuras C.4 e C.5, para o caso de uma estrutura composta por um quadrado de dimensão unitária e um triângulo equilátero. Com as arestas 1 e 2 selecionadas, acionou-se o botão destacado na Figura C.4, abrindo-se o diálogo para definição do número de divisões destas arestas. Após o aceite, os pontos que delimitam as divisões das arestas ficam em destaque para visualização.



**Figura C.4:** Mapeamento Transfinito: Divisão das arestas.

Nesta etapa, a discretização ainda não ocorreu, podendo-se modificar as divisões das arestas. Quando as divisões estiverem satisfatórias, aciona-se o botão de discretização destacado na Figura C.5 para geração da malha.

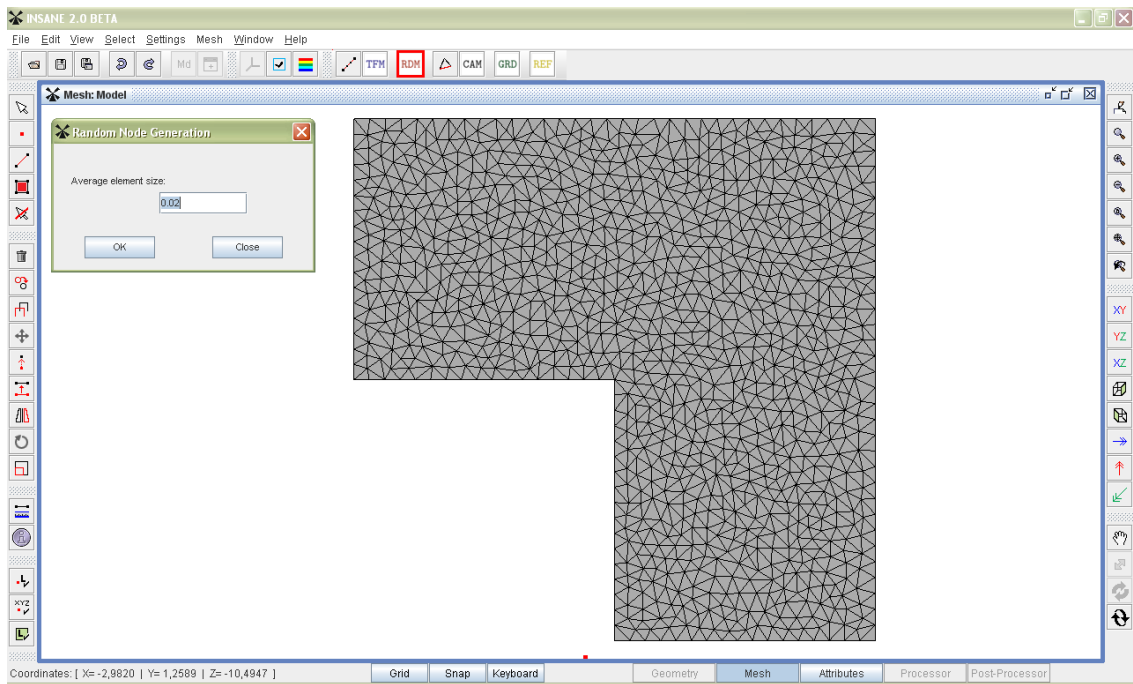


**Figura C.5:** Mapeamento Transfinito: Discretização.

Observa-se nas Figuras C.4 e C.5 que, apesar de definidos somente o número de divisões das arestas 1 e 2, as demais arestas também foram divididas tal que a malha gerada fosse conforme.

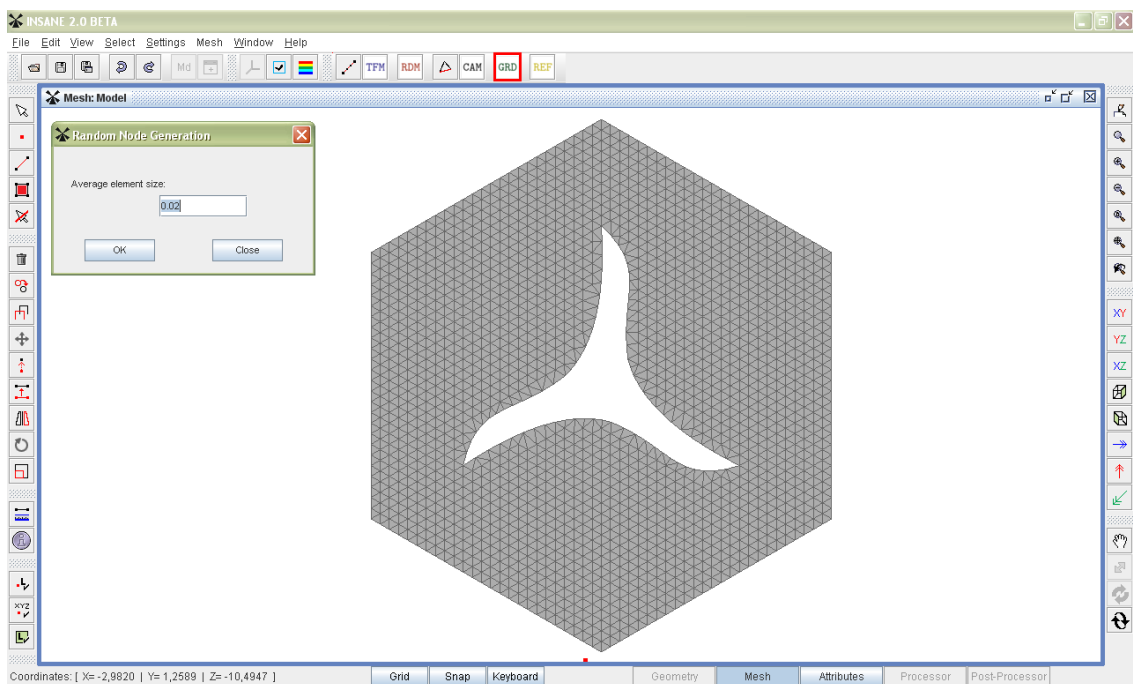
Como ilustrado na Figura C.5, regiões triangulares geram elementos triangulares, ao passo que regiões quadrilaterais geram elementos quadrilaterais. Da forma como foi implementada, esta técnica de discretização é restrita às estruturas cuja geometria possa ser decomposta em regiões triangulares e/ou quadrilaterais, com lados retos ou não. As técnicas de decomposição de domínio implementadas são alternativas para o caso de estruturas com geometrias mais complexas.

Nos casos de Geração Nodal Aleatória e Malha com base em Grade, o processo de discretização é realizado definindo-se a dimensão média dos elementos. Na Figura C.6, ilustra-se o processo de discretização pela Geração Nodal Aleatória para o caso de uma estrutura com geometria em “L” de lado unitário. Acionando-se o botão destacado na Figura C.6, abre-se o diálogo para definição da dimensão média dos elementos. Após o aceite, a malha é gerada.



**Figura C.6:** Geração Nodal Aleatória: Discretização.

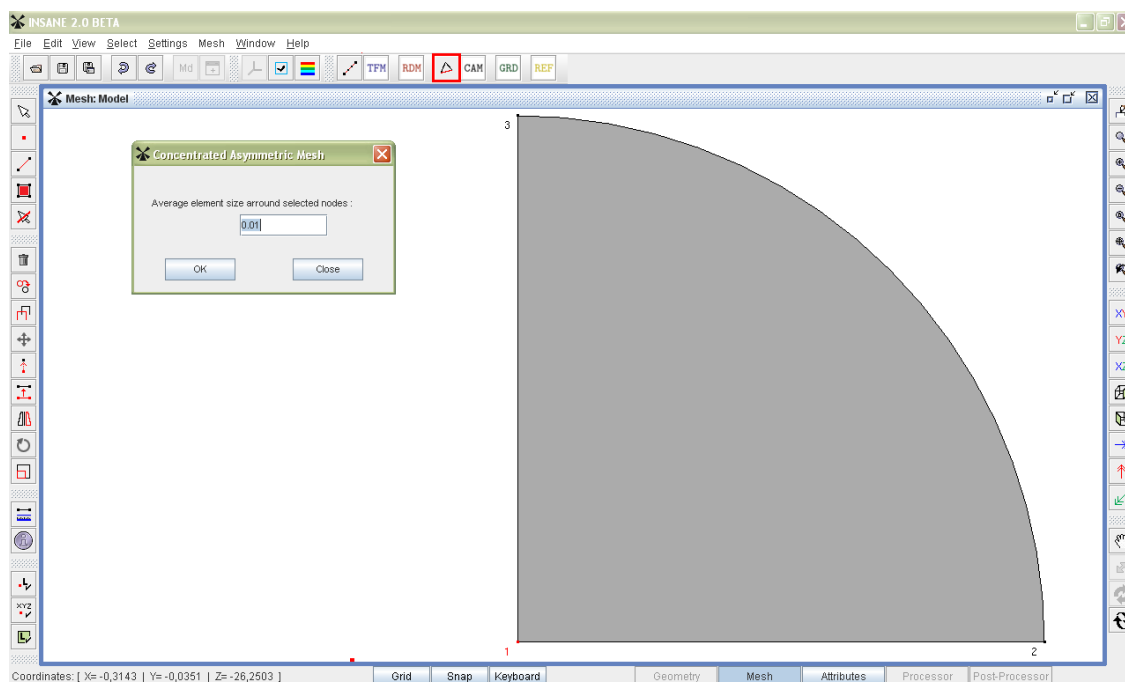
Analogamente, na Figura C.7, ilustra-se o processo de discretização pela Malha com base em Grade para o caso de uma estrutura vazada com geometria hexagonal de lado unitário.



**Figura C.7:** Malha com base em Grade: Discretização.

Fundamentalmente, a Geração Nodal Aleatória e a Malha com base em Grade se distinguem pelo algoritmo gerador da malha e, conseqüentemente, pela malha gerada. No caso de Malha com base em Grade, tem-se as opções de algoritmo gerador quadrilateral e hexagonal. Destaca-se que a discretização pela Malha com base em Grade gera sempre a mesma malha para um dado modelo geométrico, dimensão média dos elementos e algoritmo gerador, o que não é verdade para o caso de Geração Nodal Aleatória.

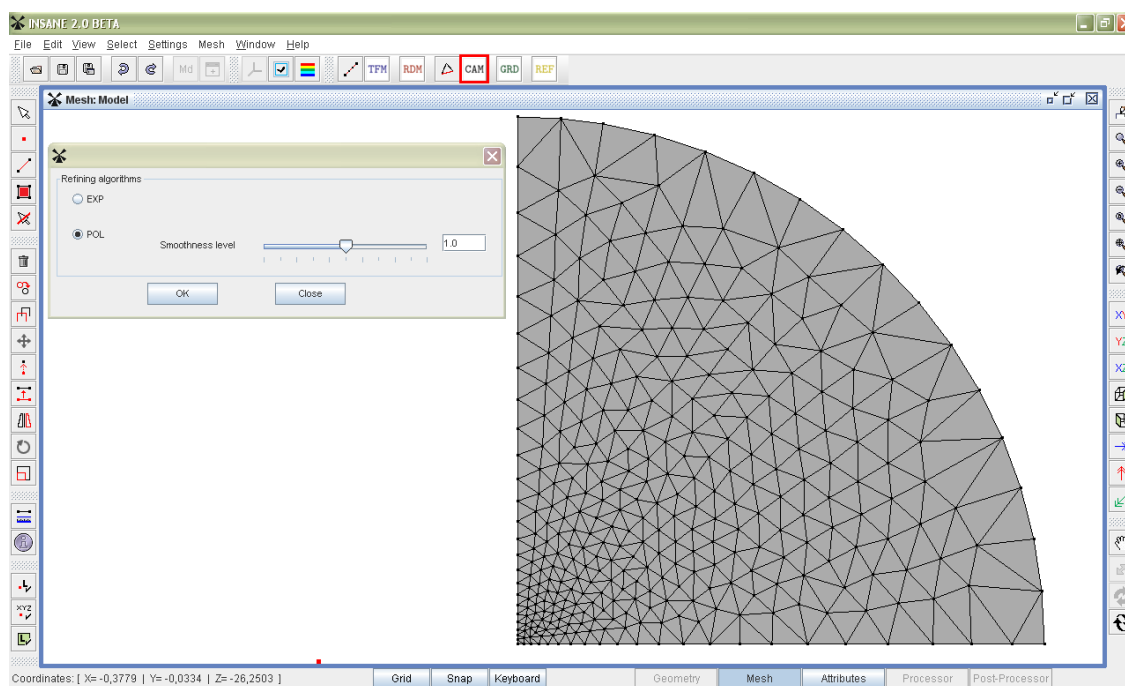
No caso de Malha Assimétrica Concentrada, o processo de discretização é análogo ao das demais técnicas de decomposição de domínio, porém, define-se a dimensão média dos elementos no entorno dos nós, permitindo-se um controle da concentração da malha. Este processo de discretização é ilustrado nas Figuras C.8 e C.9, para o caso de uma estrutura com geometria definida por um quarto de círculo de raio unitário. Com o nó 1 selecionado, acionou-se o botão destacado na Figura C.8, abrindo-se o diálogo para definição da dimensão média dos elementos no entorno deste nó.



**Figura C.8:** Definição da dimensão média dos elementos no entorno do nó 1.



Na sequência, repetiu-se este procedimento para os nós 2 e 3, definindo-se a dimensão média dos elementos no entorno destes nós como sendo 10 vezes maior do que daqueles no entorno do nó 1. Por fim, acionou-se o botão de discretização destacado na Figura C.9 para definição do algoritmo gerador entre as opções exponencial e polinomial, além do nível de suavização no caso do algoritmo gerador polinomial. Após o aceite, a malha é gerada.

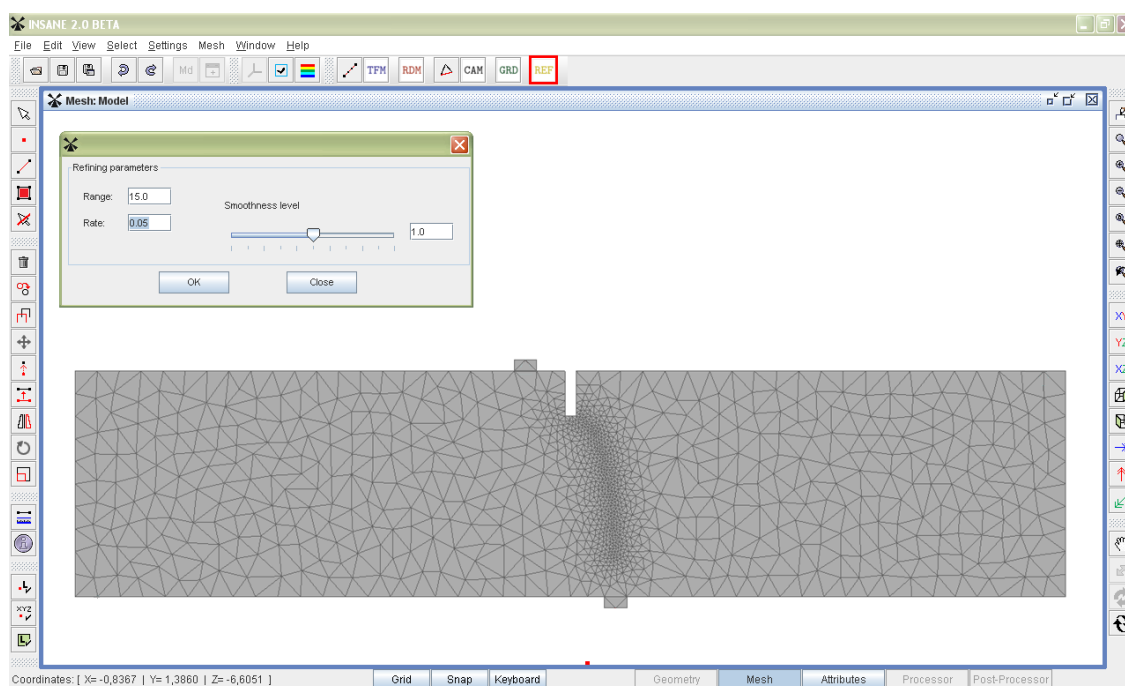


**Figura C.9:** Discretização com concentração de malha.

Conforme ilustrado nas Figuras C.6, C.7 e C.9, independentemente da geometria da estrutura, as técnicas de decomposição de domínio implementadas geram elementos triangulares. Destaca-se que, diferentemente do Mapeamento Transfinito, estas técnicas não se restringem à discretização de geometrias decompostas em regiões triangulares e/ou quadrilaterais.

Um recurso útil, comum às técnicas de geração de malhas apresentadas, é o de refinamento local da malha. Este recurso permite o aumento da densidade de malha em regiões de interesse. O processo de refinamento local é realizado estabelecendo-se a região de interesse a partir de um caminho definido por seleção de nós.

Este processo é ilustrado na Figura C.10, para o caso de modelagem do ensaio de cisalhamento em quatro pontos. Objetivou-se uma maior densidade de malha na região preferencial de fissuração. Com o caminho definido pelos nós selecionados, acionou-se o botão destacado na Figura C.10, abrindo-se o diálogo para definição da largura da região de refinamento local da malha, da dimensão média dos elementos nesta região, definida em relação à daqueles antes do refinamento local, e o nível de suavização para conformação da malha. Após o aceite, o refinamento local é gerado.



**Figura C.10:** Refinamento local da malha.

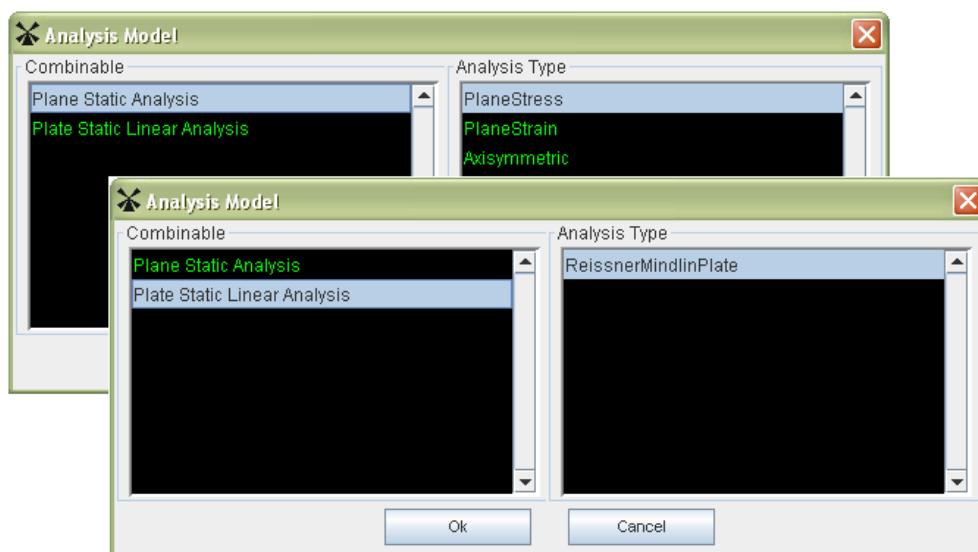
Destaca-se que, independentemente da geometria dos elementos antes do refinamento local, este processo gera elementos triangulares na região refinada.

Conforme apresentado na seção 2.4, estas técnicas de geração de malhas foram implementadas por Ferreira e Pitangueira (2015) aproveitando de implementações anteriores de Gonçalves (2004) e valendo de recursos implementados na aplicação de pós-processamento.

Neste módulo, tem-se também as barras de ferramentas laterais esquerda e direita que disponibilizam, respectivamente, recursos para seleção/edição e visualização.

Analogamente ao Módulo *Geometria*, a barra de status destaca o módulo corrente (*Malha*), bem como o botão de fluxo acionável (*Atributos*) que encapsula o comando responsável por disparar a passagem para o próximo módulo.

Definida a discretização, segue-se para o Módulo *Atributos*. Ao se acionar o botão de passagem para o Módulo *Atributos*, abre-se o diálogo para especialização da análise. No caso de análise bidimensional, opta-se entre análise estática com carregamento no plano, com as opções de análise em *Estado Plano de Tensão*, *Estado Plano de Deformação* e *Sólido Axissimétrico*, e análise estática linear com carregamento normal ao plano (placas), condicionada à teoria de *Placas de Reissner Mindlin*, conforme apresentado na Figura C.11. Da perspectiva da implementação, conforme abordado no capítulo 3, neste instante, define-se qual a classe especializada da classe abstrata *Combinable* irá gerenciar a continuidade da análise.

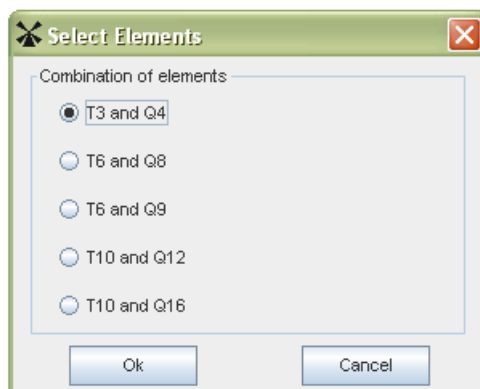


**Figura C.11:** Diálogo de definição do *Combinable* e do modelo de análise.

A opção *PlaneStaticAnalysis* disponibiliza recursos e comandos para análise pelo MEF ou pelo MEFMG de problemas bidimensionais estáticos lineares ou fisicamente não lineares com carregamento no plano, ao passo que a opção *PlateStaticLinearAnalysis* disponibiliza recursos e comandos para análise pelo MEF de problemas bidimensionais estáticos lineares com carregamento normal ao plano (placas).

Conforme capítulo 3, a classe *Combinable* é responsável por especializar o Módulo *Atributos*, definindo, por exemplo, comandos, modelos constitutivos e processos de solução, ao passo que o modelo de análise auxilia na construção de diversos diálogos, por exemplo, ao definir os graus de liberdade nodal, necessários para construção dos diálogos de aplicação das cargas e de determinação dos atributos nodais, bem como ao ser investigado para filtrar as propriedades das degenerações e dos materiais.

Após especialização da análise, abre-se o diálogo para definição dos elementos finitos a se adotar. No estágio atual do sistema, são permitidos *T3*, *T6* e *T10*, aplicáveis às faces triangulares, e *Q4*, *Q8*, *Q9*, *Q12* e *Q16*, aplicáveis às faces quadrilaterais. Destaca-se que face triangular compõe elemento finito triangular, ao passo que face quadrilateral compõe elemento finito quadrilateral. Logo, as possíveis combinações de elementos finitos, tal como ofertadas no diálogo apresentado na Figura C.12, são condicionadas à existência de faces triangulares e quadrilaterais na malha. Estas combinações são definidas pelo programa respeitando o critério da continuidade inerente ao método discreto.

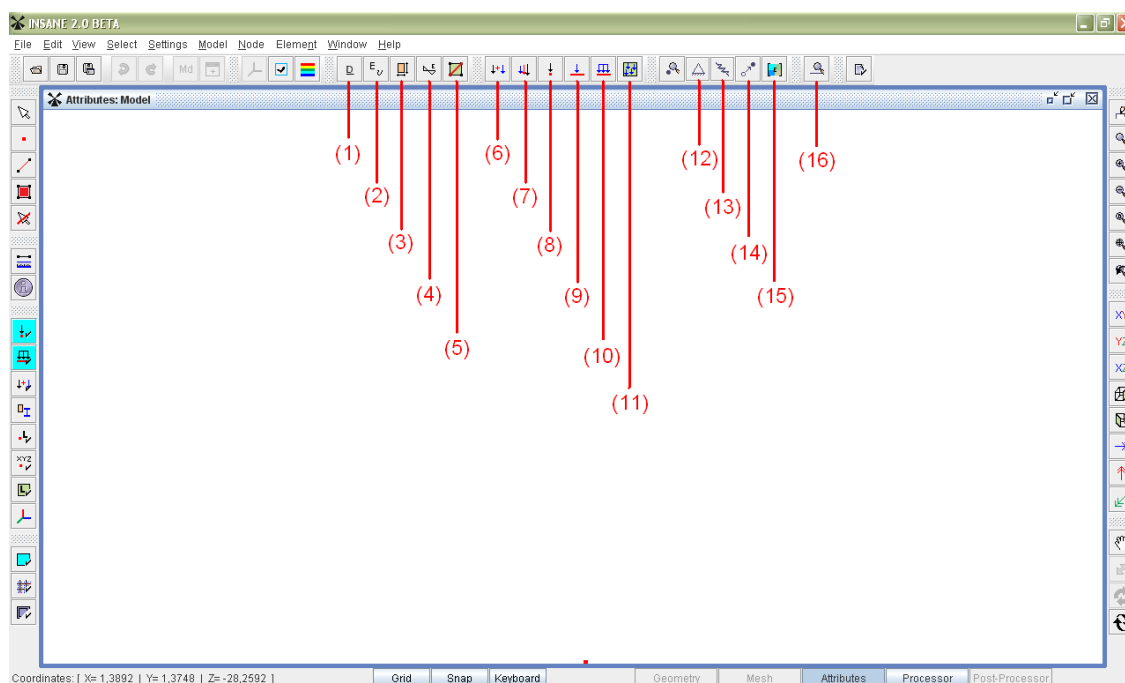


**Figura C.12:** Diálogo de definição dos elementos finitos.

Definidos os elementos finitos, abre-se o Módulo *Atributos*. Este fluxo pode ocorrer para o número de vezes de interesse do analista, cumulando janelas deste módulo cujas estruturas de dados associadas, que são modificadas quando da criação dos elementos finitos, são independentes entre si, devido à clonagem da estrutura de dados na mudança de módulos.

## C.3 Módulo *Atributos*

O Módulo *Atributos*, apresentado na Figura C.13, é composto, além dos recursos para seleção e visualização, de recursos para definição dos atributos do modelo. Neste módulo, definem-se o modelo constitutivo, os materiais, as degenerações e as condições de contorno, além de outros atributos opcionais. Quanto aos tipos de carga, permitem-se cargas nodais, cargas concentradas e linearmente distribuídas em qualquer posição das arestas dos elementos e cargas de área constantes em toda a face dos elementos.



**Figura C.13:** Pré-processor gráfico interativo: Módulo *Atributos*.

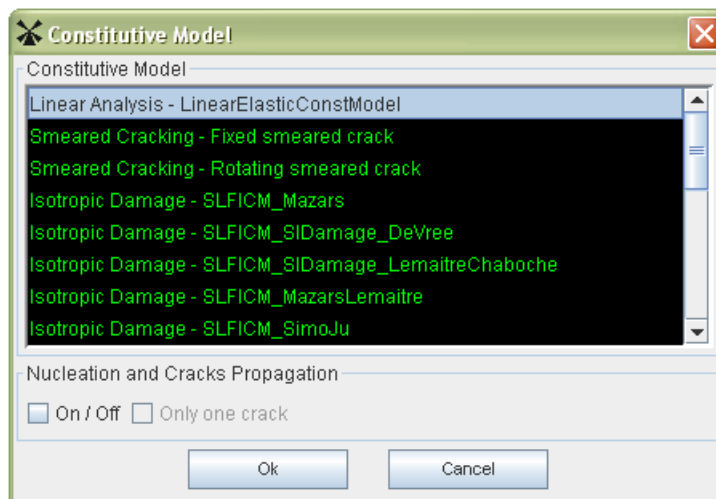
A barra de ferramentas superior disponibiliza os recursos para definição dos atributos do modelo, cujos principais são apresentados a seguir, conforme destacados na Figura C.13:

### (1) Definição do modelo constitutivo

No estágio atual do sistema, para a opção de análise com carregamento no plano, define-se o modelo constitutivo dentre as seguinte opções:

- Modelo linear elástico;
- Modelo de fissuras distribuídas com direção de fissuração fixa;
- Modelo de fissuras distribuídas com direção de fissuração rotacional;
- Modelo de dano isotrópico de Mazars (1984);
- Modelo de dano isotrópico de de Vree et al. (1995);
- Modelo de dano isotrópico de Lemaitre e Chaboche (1990);
- Modelo de dano isotrópico de Mazars e Lemaitre (1984);
- Modelo de dano isotrópico de Simo e Ju (1987);
- Modelo de dano isotrópico de Ju (1989);
- Modelo de dano isotrópico de Mazars (1984) modificado;
- Modelo de dano distribuído por de Borst e Gutierrez (1999);
- Modelo elastoplástico com dano de Lemaitre (1985*a,b*);
- Modelo elastoplástico com critério de escoamento de von Mises;
- Modelo elastoplástico com critério de escoamento de von Mises e endurecimento misto;
- Modelo elastoplástico com critério de escoamento de Drucker Prager;
- Modelo elastoplástico com critério de escoamento de Rankine;
- Modelo de dano volumétrico por Penna (2011), com leis de evolução de dano para tração e para compressão;
- Modelo de dano volumétrico por Penna (2011), com leis de evolução de dano para tração, para compressão e para deformação desviadora.

Esta definição é realizada através do diálogo apresentado na Figura C.14, onde destaca-se também a opção de a análise avaliar o processo de nucleação e propagação de trincas, abordada em detalhes a partir do capítulo 5.



**Figura C.14:** Diálogo de definição do modelo constitutivo.

As diversas opções para modelar a degradação do meio material foram abordadas no contexto de uma estrutura teórica e computacional para modelos constitutivos baseados em degradação elástica e implementadas neste arcabouço desenvolvido por Penna (2011), onde encontram-se maiores detalhes sobre este assunto.

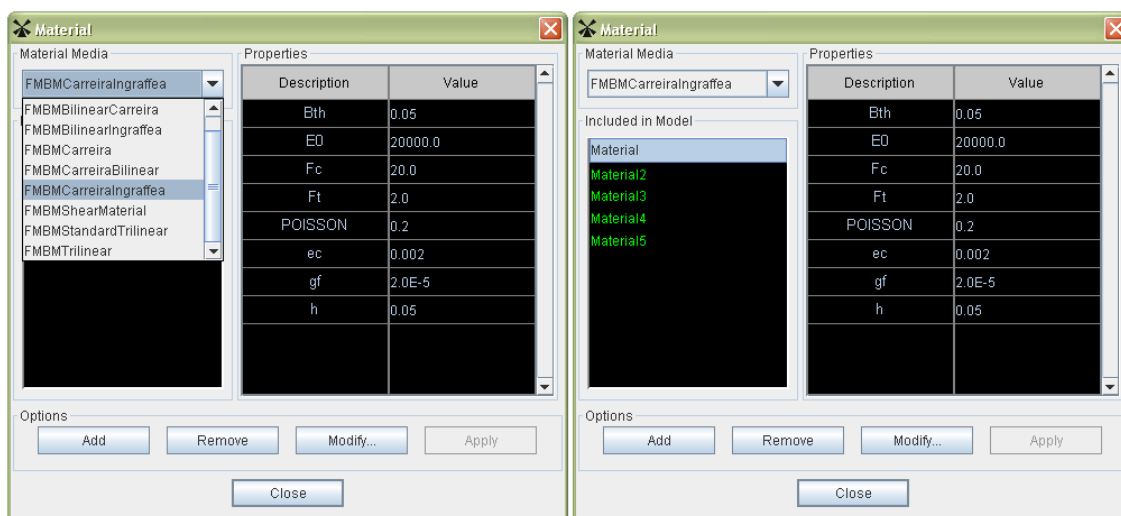
A definição do modelo constitutivo no Módulo *Atributos* permite que o modelo geométrico discretizado e com as degenerações e condições de contorno definidas seja analisado com diferentes modelos constitutivos. Para tanto, basta que se redefina o modelo constitutivo, bem como os materiais, tendo em vista que os tipos de material aplicáveis à modelagem variam em função do modelo constitutivo.

Ao se definir o modelo constitutivo, o programa cria um material com parâmetros exemplificativos e o associa à degeneração criada na construção da classe *Combinable*.

Para a opção de análise com carregamento normal ao plano (placas), não cabe ao analista a definição do modelo constitutivo, condicionado ao modelo linear elástico, sendo este recurso suprimido da interface com o usuário.

## (2) Criação de materiais

Conforme destacado, os tipos de material aplicáveis à modelagem são definidos em função do modelo constitutivo. Sendo assim, condiciona-se o acesso ao diálogo de criação de materiais à definição do modelo constitutivo. Este diálogo é apresentado na Figura C.15, cujos tipos de material apresentados são os aplicáveis aos modelos de fissuras distribuídas.



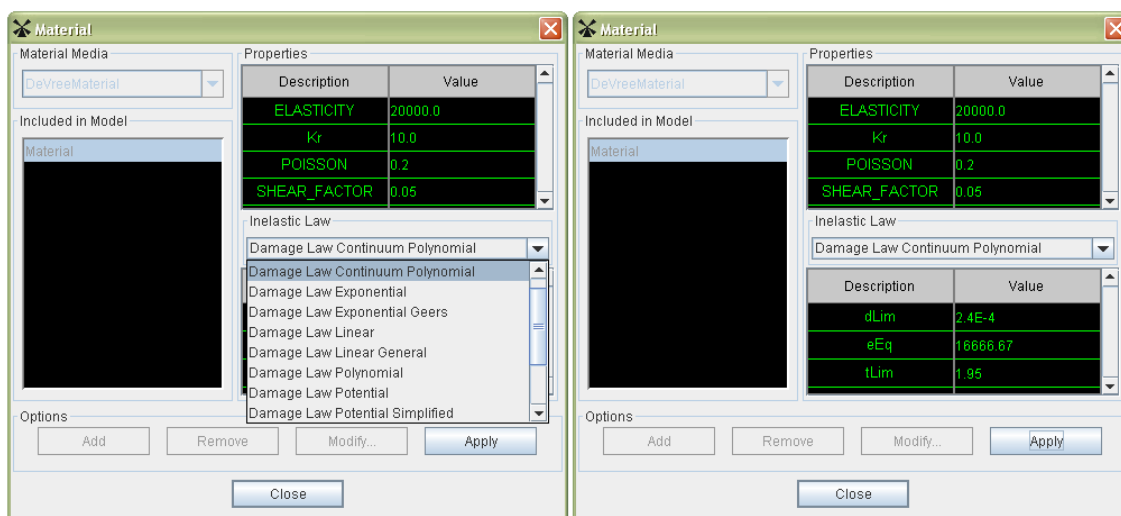
**Figura C.15:** Diálogo de criação de materiais.

O processo de criação de material é realizado escolhendo-se, primeiramente, o tipo de material na caixa de definição do meio material. Na sequência, aciona-se o botão adicionar. Após a definição do nome do material, definem-se os seus parâmetros no painel de propriedades e, por fim, aciona-se o botão aplicar. O material criado pode ainda ser modificado ou removido através dos respectivos botões.

Este diálogo contém também um painel para apresentação dos materiais incluídos no modelo, cujo selecionado tem o seu tipo e os seus parâmetros apresentados.

Determinados tipos de material requerem também a definição de leis de evolução de dano. O processo de criação de material com uma lei de evolução de dano como um de seus parâmetros é apresentado na Figura C.16, cujo tipo de material apresentado é aplicável ao modelo de dano isotrópico de de Vree et al. (1995).





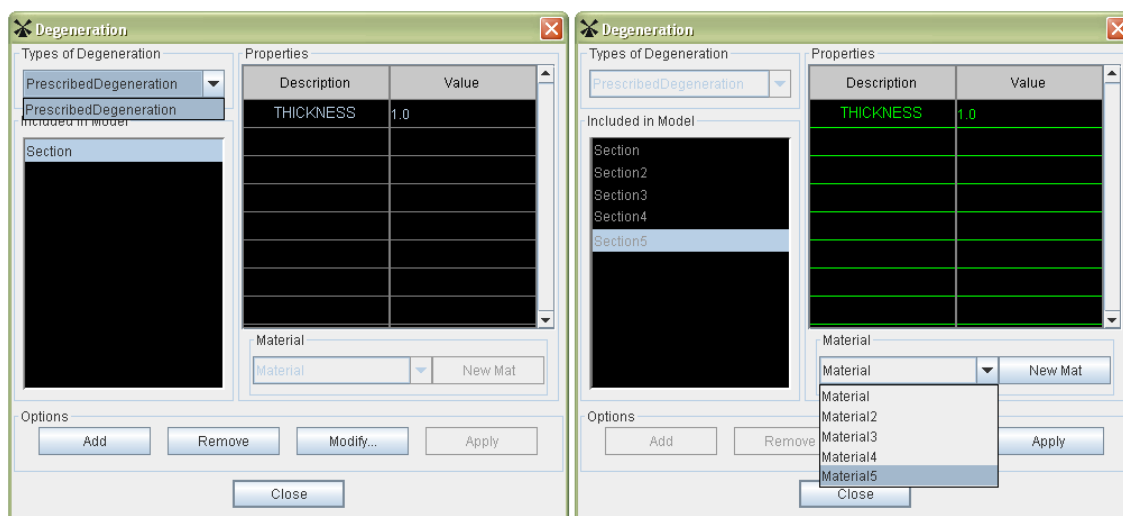
**Figura C.16:** Material com uma lei de evolução de dano como um de seus parâmetros.

As adaptações deste diálogo são automáticas em função do tipo de material requerer ou não a definição de leis de evolução de dano, avaliado através da classe *Combinable*. As leis de evolução de dano aplicáveis à modelagem são definidas em função do tipo de material.

A lógica implementada para abastecer o diálogo de criação de materiais permite que, na introdução de um novo tipo de material ou de uma nova lei de evolução de dano no núcleo numérico do sistema, esta opção seja automaticamente carregada para este diálogo. Para tanto, basta que se definam os modelos constitutivos ou os tipos de material aplicáveis nas respectivas classes. Esta propriedade é chamada de *Interfaces de Comunicação para Combinação de Recursos do Sistema INSANE*.

### (3) Criação de degenerações

O processo de criação de degeneração é análogo ao de criação de material. Através do diálogo apresentado na Figura C.17, primeiramente, escolhe-se o tipo de degeneração e, em seguida, aciona-se o botão adicionar. Após a definição do nome da degeneração, definem-se as suas propriedades e, por fim, aciona-se o botão aplicar. A degeneração criada também pode ser modificada ou removida através dos respectivos botões.



**Figura C.17:** Diálogo de criação de degenerações.

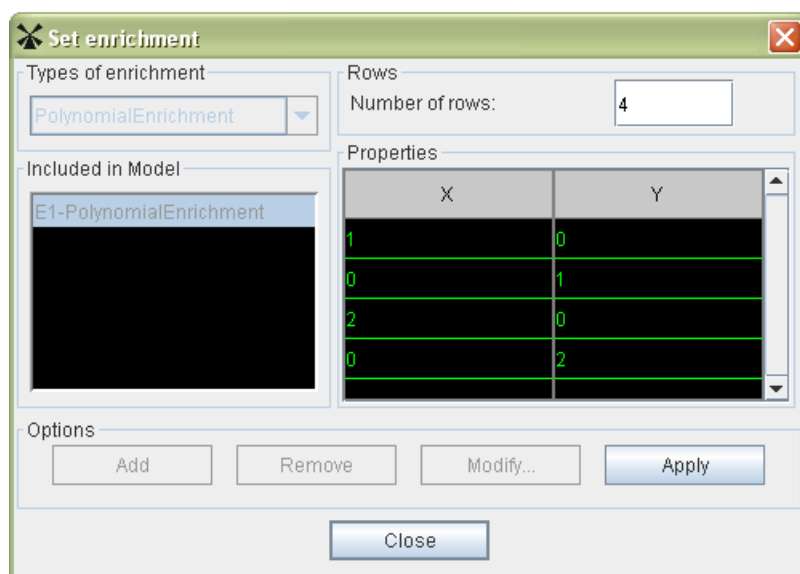
Na Figura C.17, destaca-se o processo de associação de um material à degeneração, realizado quando da definição das propriedades desta degeneração.

O diálogo de criação de degenerações contém também um painel para apresentação das degenerações incluídas no modelo, cuja selecionada tem o seu tipo e os seus parâmetros apresentados.

#### (4) Criação de funções de enriquecimento nodal de MEFG

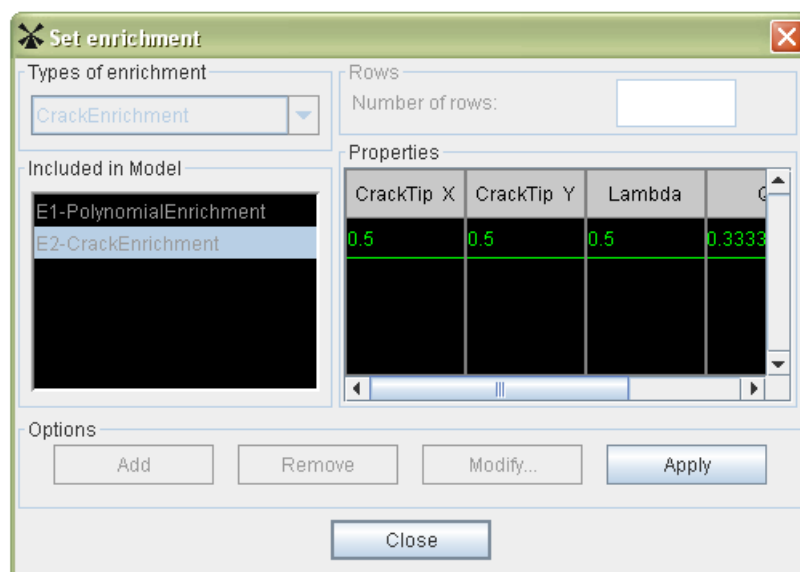
O processo de criação de função de enriquecimento nodal é análogo aos de criação de material e de degeneração. No ambiente gráfico, permitem-se os enriquecimentos nodais com funções polinomiais e de singularidade. Enriquecimentos global-local e com funções trigonométricas estão em fase de desenvolvimento.

As funções polinomiais são criadas definindo-se os seus monômios. Através do diálogo apresentado na Figura C.18, primeiramente, define-se o número de linhas da tabela no correspondente painel e aperta-se *enter*, necessário para adequação desta tabela onde cada linha refere-se a um monômio. Na sequência, definem-se os expoentes dos componentes de cada um dos monômios. No caso de análise bidimensional, esta entrada consiste em definir os expoentes  $p$  e  $q$  de cada um dos monômios  $x^p y^q$  que compõem a função de enriquecimento polinomial.



**Figura C.18:** Diálogo de criação de funções de enriquecimento polinomial.

O enriquecimento nodal com função de singularidade é empregado para simular a ocorrência deste fenômeno em pontas de trinca. Através do diálogo apresentado na Figura C.19, cria-se este enriquecimento definindo-se os parâmetros geométricos da trinca, dentre outros parâmetros obtidos a partir da Mecânica da Fratura abordados em maiores detalhes em Alves (2012).

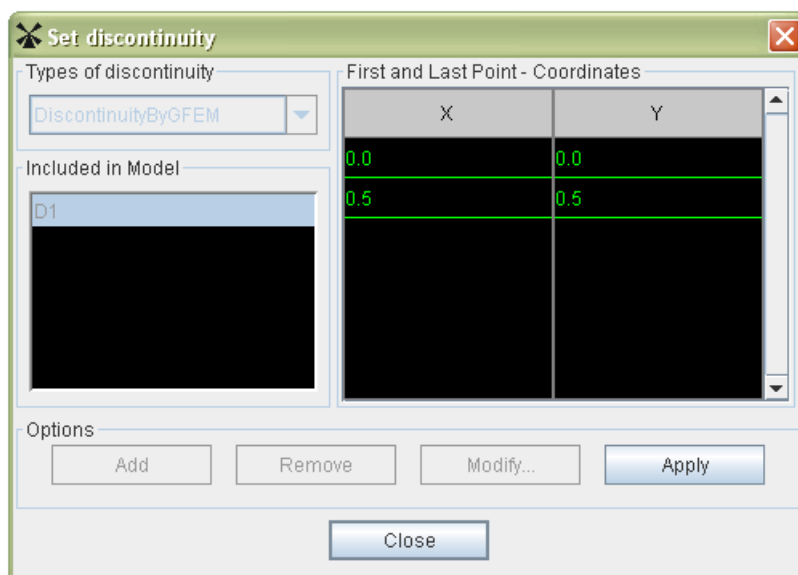


**Figura C.19:** Diálogo de criação de funções de enriquecimento de singularidade.

Estas funções de enriquecimento nodal foram implementadas em um arcabouço computacional para análise pelo MEFG desenvolvido por Alves (2012), onde encontram-se maiores detalhes sobre este assunto. Neste trabalho, aborda-se também a estratégia de enriquecimento global-local implementada no núcleo numérico do sistema INSANE e em fase de estudo para ser disponibilizada no sistema gráfico integrado.

### (5) Criação de descontinuidades (entalhes), simuladas pelo MEFG

Este recurso permite a modelagem de descontinuidades preexistentes na estrutura sem a necessidade de introduzi-las na geometria do modelo. Através do diálogo apresentado na Figura C.20, cria-se uma descontinuidade definindo-se as coordenadas de seus pontos inicial e final.



**Figura C.20:** Diálogo de criação de descontinuidades.

A partir desta entrada, o programa se encarrega de determinar todos os elementos cortados por esta descontinuidade e de definir e enriquecer com função degrau os nós necessários para simulação do salto no campo de deslocamentos.

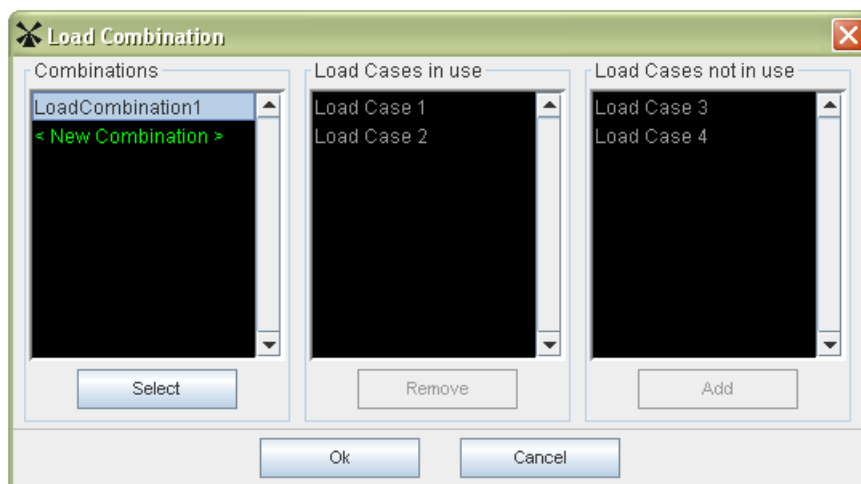
Destaca-se que estas descontinuidades são simuladas como entalhes, ou seja, sem a consideração de existência de forças de coesão atuantes em seus planos.

## (6) Criação de combinações de carregamentos

No sistema INSANE, os tipos de carga - nodal, concentrada, linear, de área, dentre outras - são agrupadas em objetos carregamentos. Cada objeto carregamento, por sua vez, é associado a um objeto caso de carregamento, através do qual definem-se o seu fator de escala e se este é incremental ou não. Por fim, os casos de carregamento são associados aos objetos de combinações de carregamentos.

No ambiente gráfico, por padrão, todos os casos de carregamento criados pelo analista são associados à combinação de carregamentos criada na construção da classe *Combinable*, porém, passíveis de desassociação.

Desta forma, o acionamento do diálogo apresentado na Figura C.21 somente é necessário se o analista quiser criar novas combinações de carregamentos ou redefinir os casos de carregamento aplicáveis.



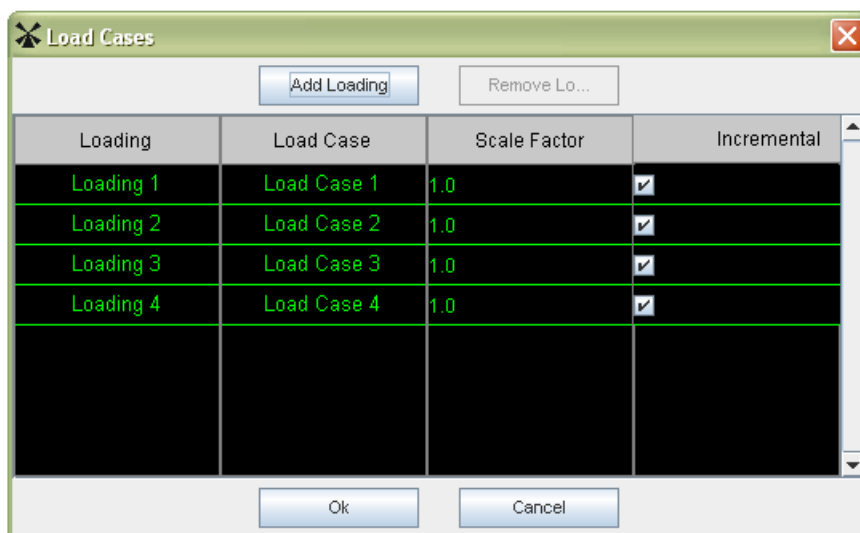
**Figura C.21:** Diálogo de criação de combinações de carregamentos.

Para a criação de combinação de carregamentos, na opção de nova combinação, aciona-se o botão selecionar.

Analogamente, para a definição dos casos de carregamento aplicáveis, na combinação de carregamentos de interesse, também aciona-se o botão selecionar. Após adicionar e/ou remover os casos de carregamento, aciona-se o botão atualizar, que aparece no lugar do botão selecionar.

### (7) Criação de casos de carregamento

Os casos de carregamento podem ser criados através do diálogo apresentado na Figura C.22, acionando-se o botão adicionar.



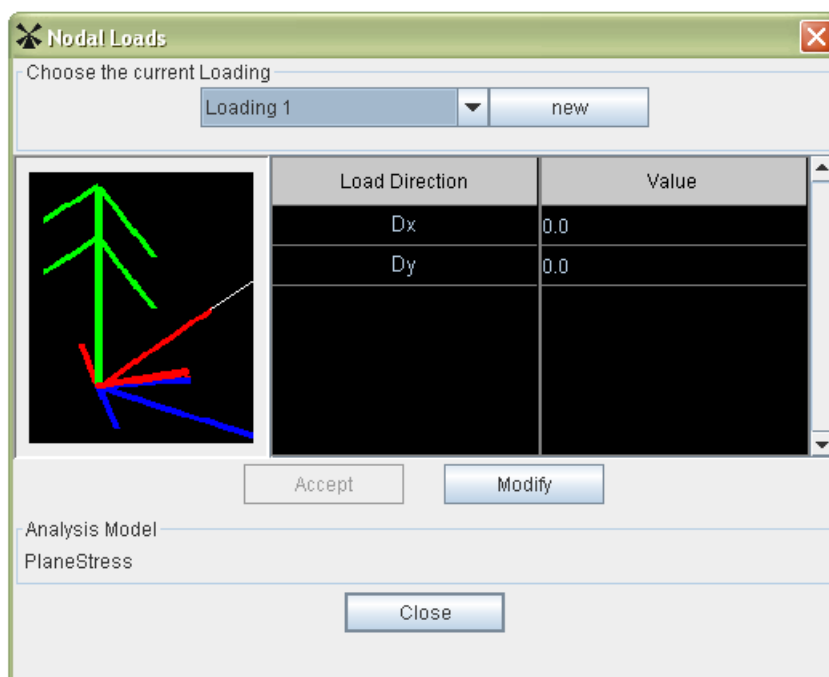
**Figura C.22:** Diálogo de criação de casos de carregamento.

Ao se criar um caso de carregamento, definem-se o fator de escala do carregamento associado e se este é incremental ou não. Por padrão, o fator de escala é inicializado com valor unitário e o carregamento é definido como não incremental se a análise for linear e como incremental se não linear.

Os casos de carregamento podem ainda ser criados implicitamente através dos diálogos de aplicação de cargas, quando da criação de carregamentos a partir destes diálogos.

### (8) Aplicação de cargas nodais

O processo de aplicação de carga nodal se inicia selecionando-se os nós de interesse. Se nenhum nó for selecionado, a carga será aplicada em todos os nós do modelo. Em seguida, através do diálogo apresentado na Figura C.23, escolhe-se ou cria-se o carregamento a que pertence esta carga. Por fim, acionando-se o botão modificar, definem-se os valores de seus componentes e aciona-se o botão de aceite. Estes componentes são estabelecidos em função dos graus de liberdade nodal.



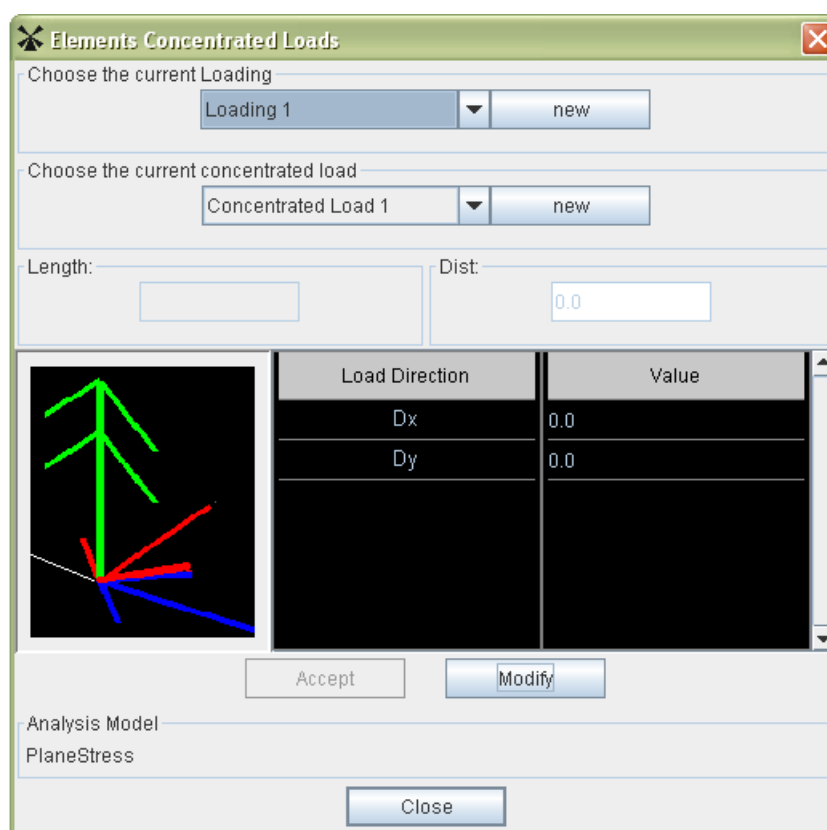
**Figura C.23:** Diálogo de aplicação de cargas nodais.

### (9) Aplicação de cargas concentradas em arestas

O processo de aplicação de carga concentrada em arestas de elementos é análogo ao de aplicação de carga nodal.

Este processo se inicia selecionando-se as arestas de interesse. Se nenhuma aresta for selecionada, a carga será aplicada em todas as arestas do modelo. Em seguida, através do diálogo apresentado na Figura C.24, escolhe-se ou cria-se o carregamento a que pertence esta carga. Diferentemente da carga nodal, a carga concentrada requer a definição de um nome para distingui-la das diversas cargas concentradas que podem ser aplicadas em uma mesma aresta. Após a definição deste nome, definem-se o ponto de aplicação da carga e os valores de seus componentes e aciona-se o botão de aceite. O ponto de aplicação da carga é definido pela sua distância ao nó inicial da aresta.

No caso de somente uma aresta selecionada, este diálogo apresenta ainda o comprimento desta aresta, com o objetivo de auxiliar na definição do ponto de aplicação da carga.



**Figura C.24:** Diálogo de aplicação de cargas concentradas em arestas.

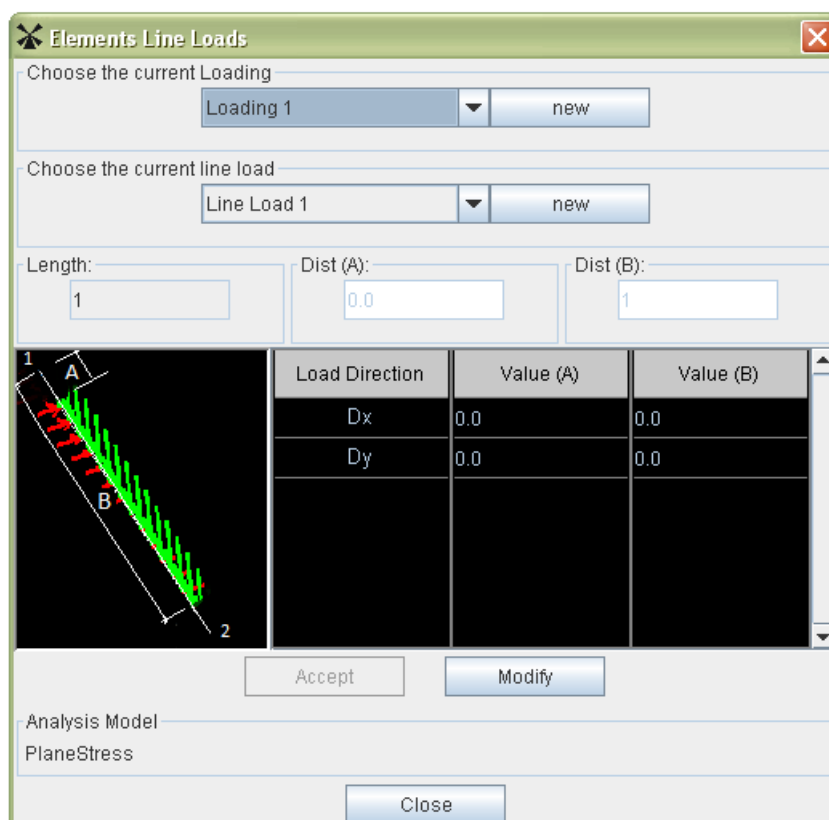
### (10) Aplicação de cargas linearmente distribuídas em arestas

O processo de aplicação de carga linearmente distribuída em arestas de elementos é análogo aos de aplicação de cargas nodal e concentrada em arestas.

Através do diálogo apresentado na Figura C.25, com as arestas de interesse selecionadas, escolhe-se ou cria-se o carregamento a que pertence esta carga. Após a definição de um nome para a carga, definem-se os pontos inicial e final de sua aplicação e os valores de seus componentes nestes pontos e aciona-se o botão de aceite. Estes pontos são definidos pelas suas distâncias ao nó inicial da aresta. Destaca-se que, no caso de mais de uma aresta selecionada, a carga distribuída será aplicada em todo o comprimento das arestas, não cabendo ao analista a definição de seus pontos inicial e final.

Analogamente ao diálogo de aplicação de cargas concentradas, no caso de somente uma aresta selecionada, este diálogo apresenta o comprimento desta aresta.





**Figura C.25:** Diálogo de aplicação de cargas linearmente distribuídas em arestas.

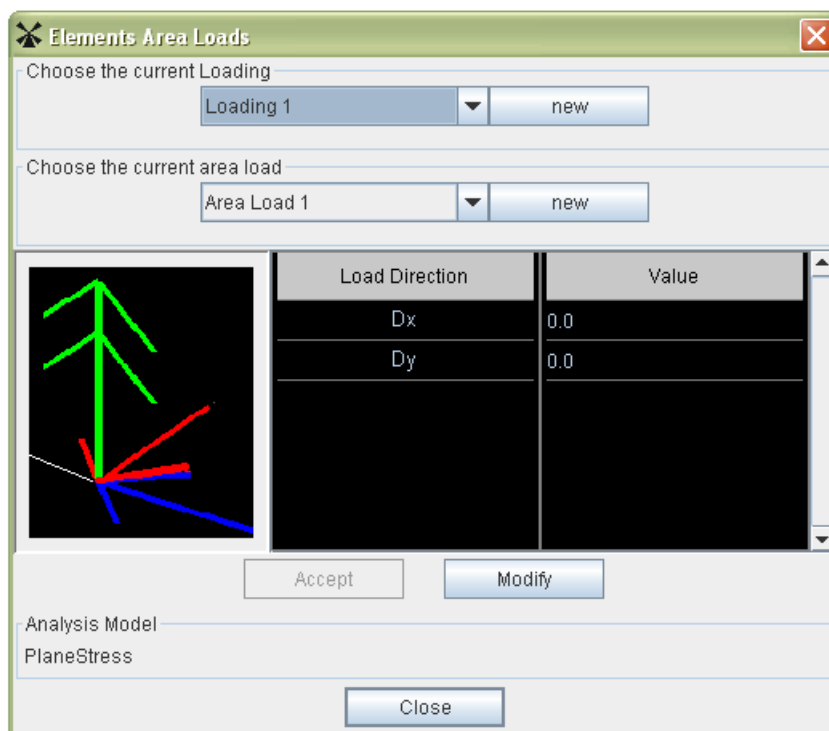
### (11) Aplicação de cargas de área

O processo de aplicação de carga de área é análogo aos demais processos de aplicação de cargas.

No ambiente gráfico, a carga de área se restringe a um valor constante aplicado em toda a face dos elementos.

Este processo se inicia selecionando-se as faces de interesse. Se nenhuma face for selecionada, a carga será aplicada em todas as faces do modelo. Em seguida, através do diálogo apresentado na Figura C.26, escolhe-se ou cria-se o carregamento a que pertence esta carga. Após a definição de um nome para a carga, definem-se os valores de seus componentes e aciona-se o botão de aceite.

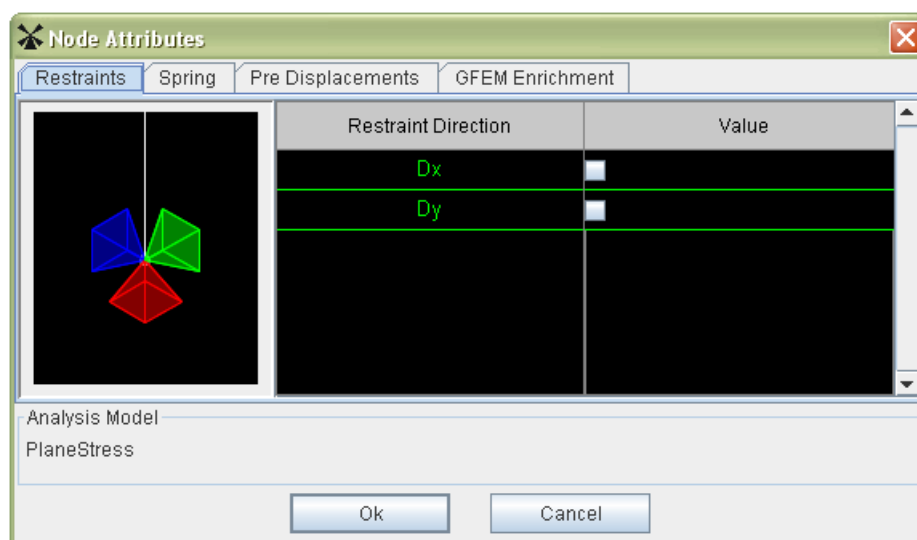
Objetivando um maior conforto visual e um menor custo computacional, as cargas de área são representadas graficamente através de um conjunto de vetores dispostos somente no entorno do centroide do elemento.



**Figura C.26:** Diálogo de aplicação de cargas de área.

## (12) Atribuição de restrições aos deslocamentos nodais

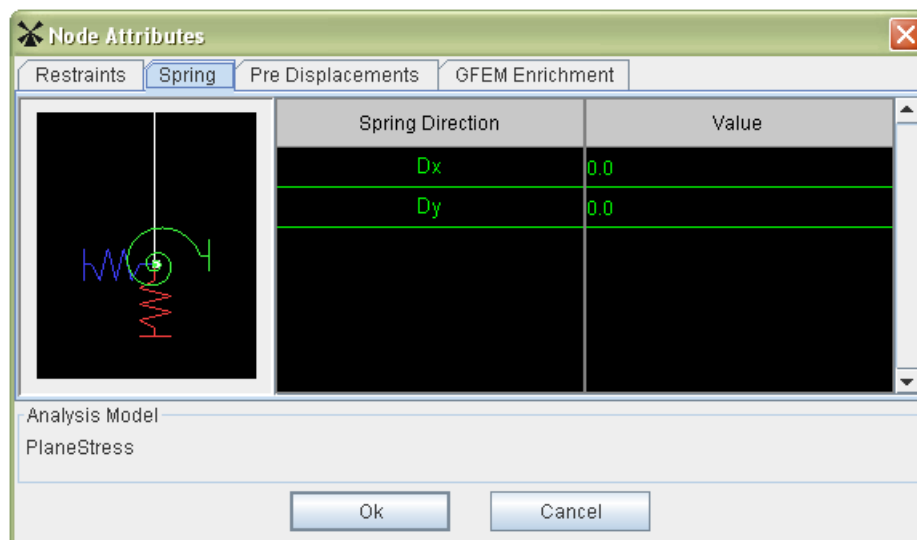
As restrições são atribuídas aos deslocamentos dos nós selecionados (ou de todos, se nenhum for selecionado) através do diálogo apresentado na Figura C.27, cujos tipos de deslocamento são os graus de liberdade nodal.



**Figura C.27:** Diálogo de atribuição de restrições aos deslocamentos nodais.

### (13) Atribuição de rigidezes de mola aos nós

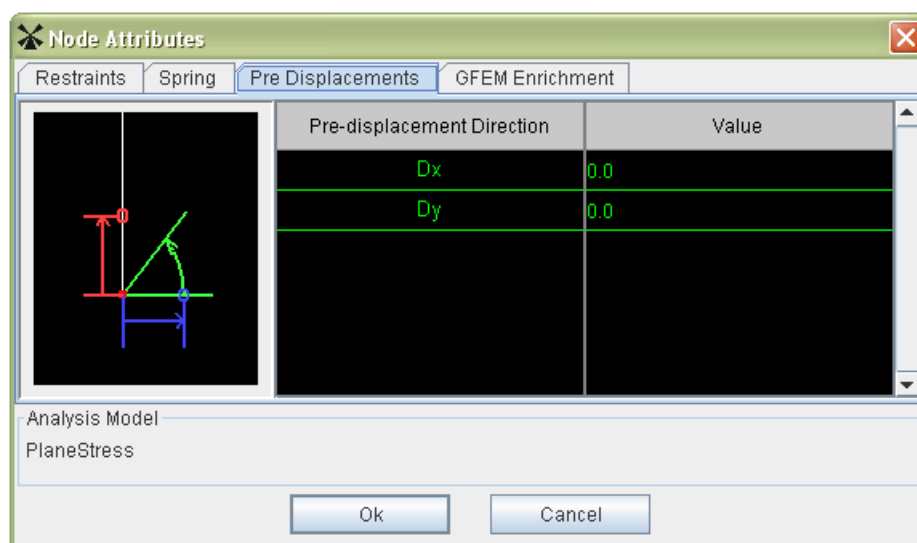
As rigidezes de mola são atribuídas aos nós analogamente às restrições. Através do diálogo apresentado na Figura C.28, definem-se seus valores.



**Figura C.28:** Diálogo de atribuição de rigidezes de mola aos deslocamentos nodais.

### (14) Atribuição de pré-deslocamentos aos nós

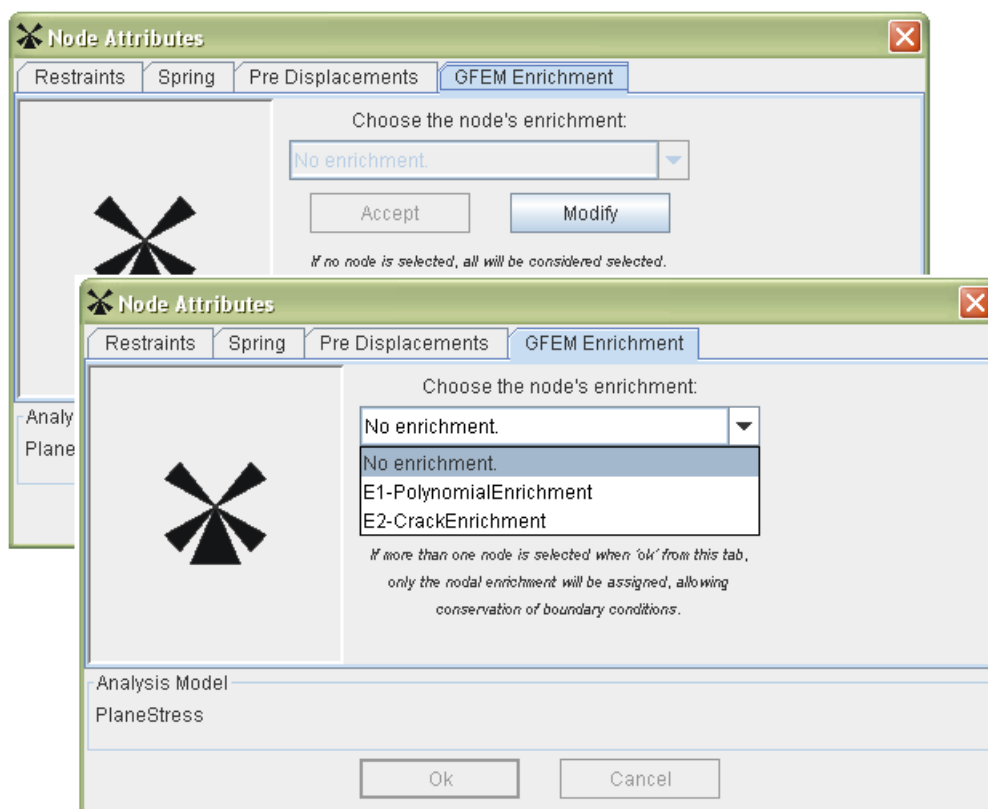
Os pré-deslocamentos são atribuídos aos nós analogamente aos demais atributos nodais. Através do diálogo apresentado na Figura C.29, definem-se seus valores.



**Figura C.29:** Diálogo de atribuição de pré-deslocamentos aos nós.

### (15) Atribuição de funções de enriquecimento aos nós

O processo de atribuição de função de enriquecimento aos nós se inicia selecionando-se os nós de interesse. Se nenhum nó for selecionado, a função de enriquecimento será atribuída a todos os nós do modelo. Em seguida, através do diálogo apresentado na Figura C.30, aciona-se o botão modificar para escolha da função de enriquecimento dentre aquelas criadas através do diálogo apresentado nas Figuras C.18 e C.19. Destaca-se a opção *sem enriquecimento*. Após a escolha, aciona-se o botão de aceite.



**Figura C.30:** Diálogo de atribuição de funções de enriquecimento aos nós.

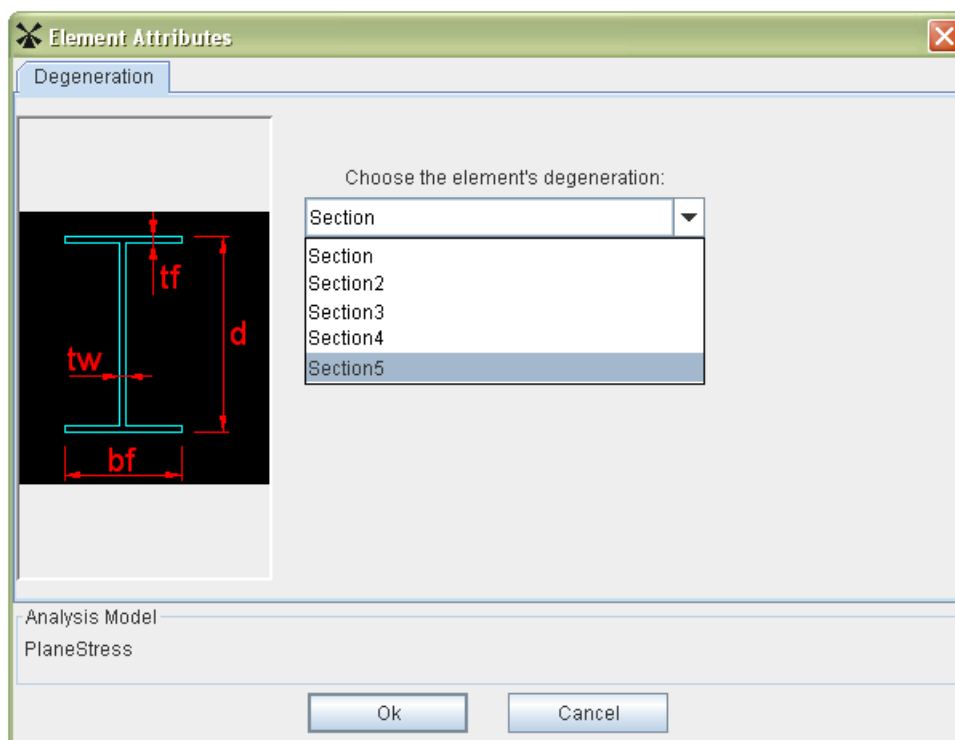
No ambiente gráfico, permite-se a atribuição de diversas funções de enriquecimento nodal, porém, somente uma por nó.

Destaca-se que os diálogos apresentados nas Figuras C.27, C.28, C.29 e C.30 consistem em abas de um mesmo diálogo, chamado diálogo de atributos nodais. Desta forma, as atribuições nodais podem ser definidas de uma só vez.

Ao se acionar o botão *Ok* do diálogo de atributos nodais, as definições em todas as abas são atribuídas aos nós selecionados, com exceção do caso em que se tem mais de um nó selecionado (ou nenhum) e este botão é acionado com a aba de atribuição de funções de enriquecimento como a corrente. Neste caso, somente a função de enriquecimento é atribuída. Isto permite a atribuição de função de enriquecimento à diversos nós simultaneamente, sem se ater aos demais atributos nodais definidos através deste diálogo. Esta lógica confere praticidade quando se analisa um modelo variando-se as funções de enriquecimento, pois não afeta as condições de contorno.

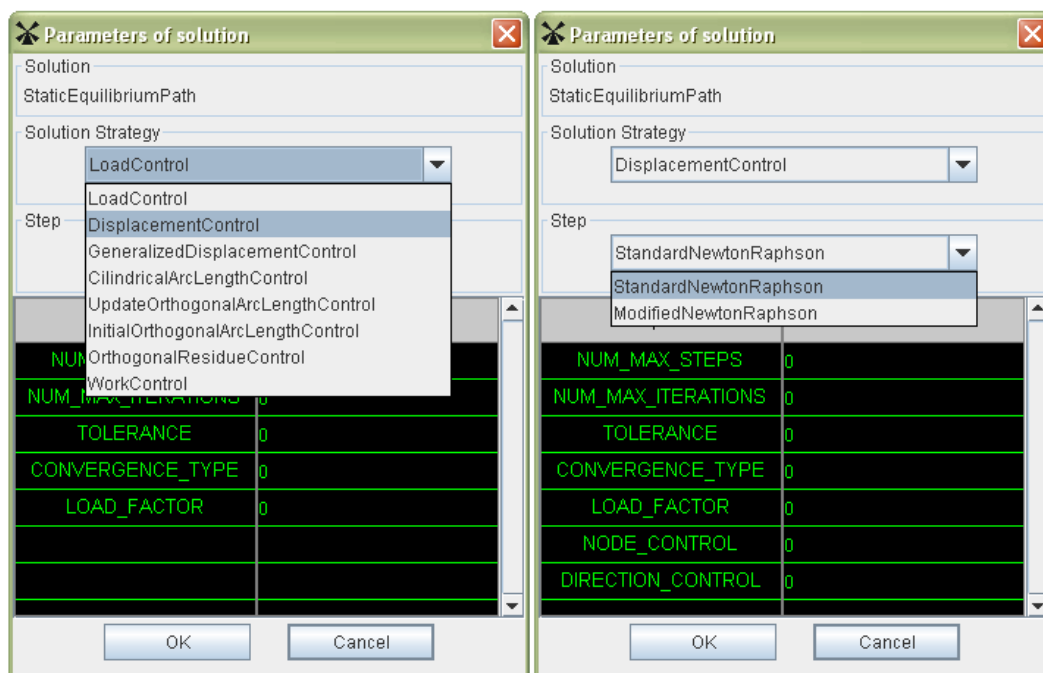
### (16) Atribuição de degenerações aos elementos

O processo de atribuição de degeneração aos elementos se inicia selecionando-se os elementos de interesse. Se nenhum elemento for selecionado, a degeneração será atribuída a todos os elementos do modelo. Em seguida, através do diálogo apresentado na Figura C.31, escolhe-se a degeneração dentre aquelas criadas através do diálogo apresentado na Figura C.17. Após a escolha, aciona-se o botão *Ok*.



**Figura C.31:** Diálogo de atribuição de degenerações aos elementos.

Além da barra de ferramentas superior, este módulo possui as barras de ferramentas laterais esquerda e direita que disponibilizam, respectivamente, recursos para seleção e visualização. Na barra de status, tem-se destacado o módulo corrente (*Atributos*), bem como liberado o botão *Processar*. Este botão encapsula o comando responsável por disparar o processamento da análise. Ao acioná-lo, primeiramente, uma análise da consistência dos dados de entrada é realizada pela classe *Combinable*. Em seguida, no caso de nenhuma inconsistência, abre-se o diálogo apresentado na Figura C.32 para definição dos parâmetros da técnica de solução, incluindo o processo incremental-iterativo e o método de controle e seus parâmetros, se aplicáveis.



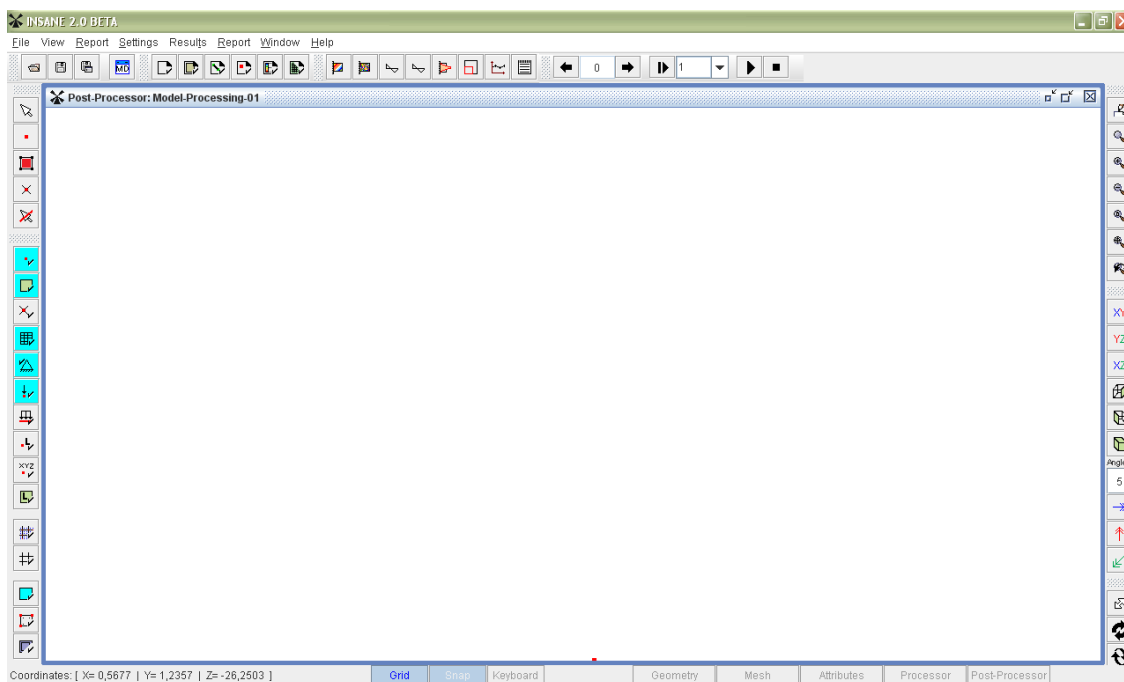
**Figura C.32:** Diálogo de definição dos parâmetros da técnica de solução.

Ao se acionar o botão *Ok*, os objetos *Model* e *Solution* são enviados ao núcleo numérico para o processamento da análise. O processamento apenas finaliza um fluxo de execução, não impedindo o retorno à etapa de pré-processamento. A cada processamento, cria-se um subdiretório *Processing-XX* no diretório definido no início do programa para armazenamento dos seus arquivos de entrada e saída, gerados em XML pelo núcleo numérico, possibilitando posterior recuperação destes dados.

Após o processamento, a barra de status libera o botão *Pós-Processador*. Este botão encapsula o comando responsável por carregar os arquivos XML dos resultados de todas as etapas do processamento para abastecimento do pós-processador.

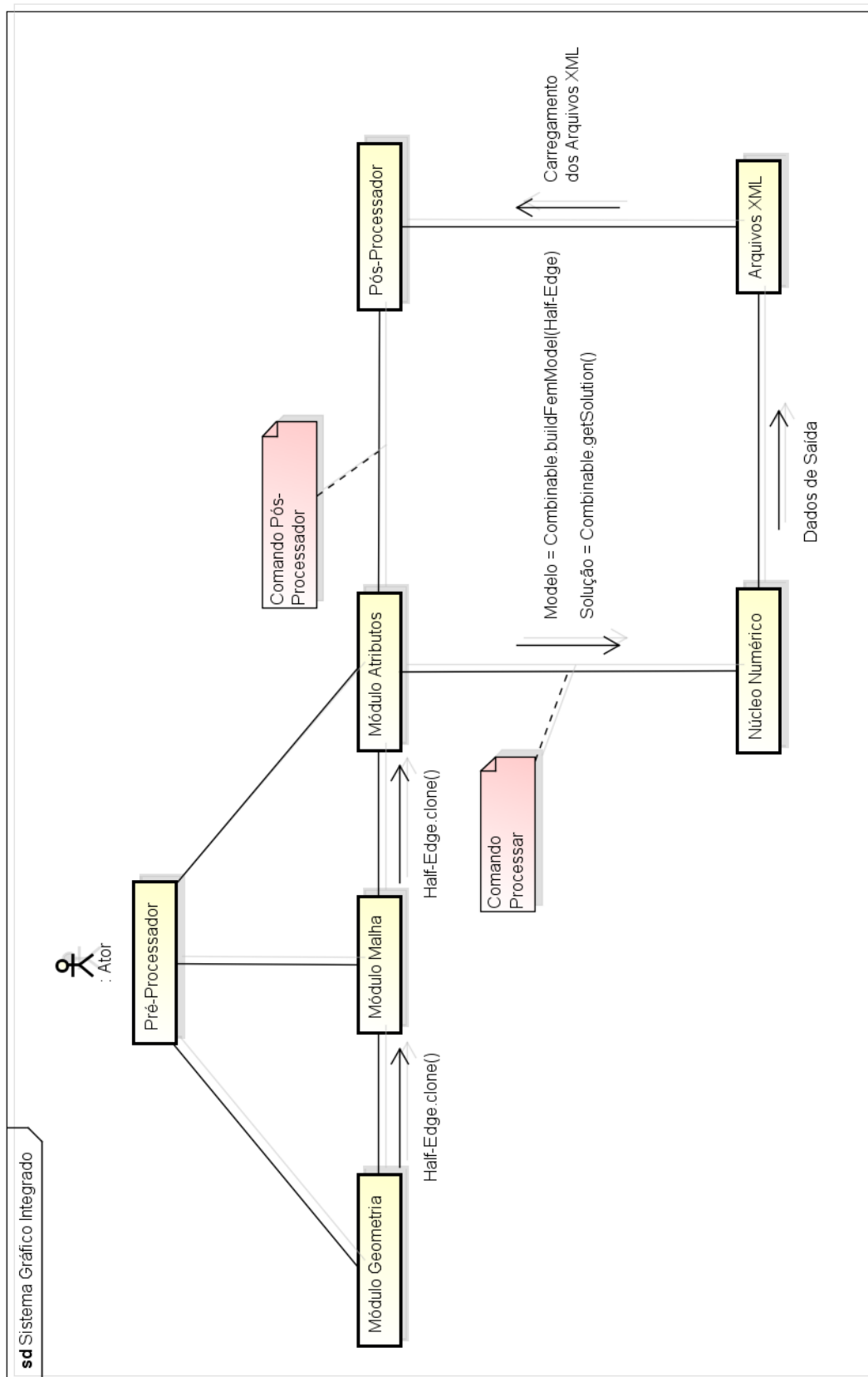
## C.4 Pós-Processador

O pós-processador, apresentado na Figura C.33, é uma aplicação gráfica interativa que disponibiliza ferramentas de visualização de resultados. Esta aplicação foi desenvolvida por Penna (2007), onde encontram-se maiores detalhes sobre sua implementação, bem como sobre seus recursos.



**Figura C.33:** Pós-processador.

O sistema gráfico integrado é apresentado esquematicamente na Figura C.34 através de um diagrama de comunicação. Destacam-se os agentes responsáveis por estabelecer a comunicação entre os módulos do pré-processador, bem como os responsáveis por estabelecer a comunicação entre esta aplicação, o processador e o pós-processador.



**Figura C.34:** Diagrama de comunicação do sistema gráfico integrado.



# Apêndice D

## Classe *DiscontinuityByGFEM*

No capítulo 8, apresenta-se o Projeto Orientado a Objetos implementado no Núcleo Numérico do INSANE para modelagem de Nucleação e Propagação de Trincas coesivas em meios bidimensionais, com enfoque no projeto *DiscontinuityByGFEM*, responsável pela simulação e gerenciamento de trincas via abordagem de enriquecimento nodal. As variáveis da classe *DiscontinuityByGFEM* armazenam os objetos necessários ao gerenciamento de trincas e entalhes. Estas variáveis são apresentadas a seguir:

- `discreteConstitutiveModel`: modelo constitutivo discreto;
- `elemDiscontinuity`: lista dos elementos cortados pela descontinuidade;
- `elements`: lista dos elementos do modelo;
- + `ELM_FIRST_POINT`: chave que se associa ao nome da descontinuidade para armazenamento do primeiro ponto de um segmento no mapa *elementValues* do elemento que o contém;
- + `ELM_SECOND_POINT`: chave que se associa ao nome da descontinuidade para armazenamento do segundo ponto de um segmento no mapa *elementValues* do elemento que o contém;
- `enrichedNodes`: lista dos nós enriquecidos devido à descontinuidade;

- enrichment: enriquecimento do tipo degrau associado à descontinuidade;
- firstElement: elemento que contém o primeiro ponto da descontinuidade;
- firstPoint: primeiro ponto da descontinuidade;
- label: nome da descontinuidade;
- lastElement: elemento que contém o último ponto da descontinuidade;
- lastPoint: último ponto da descontinuidade;
- model: objeto *Model*.

A classe *DiscontinuityByGFEM* é composta por construtores específicos à aplicação que se destina. Estes construtores são apresentados a seguir:

- + *DiscontinuityByGFEM*(): construtor vazio;
- + *DiscontinuityByGFEM*(*IPoint3d*, *IPoint3d*, *String*, *Model*): construtor de entalhes. Responsável por disparar o método *buildNotch*. Recebe como argumentos os pontos inicial e final que definem o entalhe, o nome a este atribuído e o objeto *Model*;
- + *DiscontinuityByGFEM*(*String*, *Model*, *Element*): construtor de trincas. Responsável por disparar o método *buildCrack*. Recebe como argumentos o nome para a trinca, definido pelo programa, o objeto *Model* e o elemento onde se identificou o fenômeno de nucleação de fissuras.

Os métodos da classe *DiscontinuityByGFEM* gerenciam a representação de trincas e entalhes. Estes métodos são apresentados a seguir:

- *buildCrack*(*Element*): responsável pela construção de uma trinca a partir da identificação do fenômeno de nucleação de fissuras. Recebe como argumento o elemento onde se identificou este fenômeno;

- `buildNotch()`: responsável pela construção de um entalhe a partir dos dois pontos que o definem. Este método identifica os elementos cortados por esta descontinuidade reta e define e enriquece os nós necessários para simulação do salto no campo de deslocamentos. Ressalta-se que não se atribui coesão aos entalhes;
- `enrichingNodes(ArrayList<Node>)`: responsável por enriquecer com a função de enriquecimento do tipo degrau associada à descontinuidade os nós armazenados na lista recebida como argumento, bem como por adequar as grandezas destes nós aos graus de liberdade adicionais;
- + `getAllPoints()`: retorna uma lista sequencial dos pontos que definem a descontinuidade;
- `getDiscontinuityDirectionByNonLocalPrincipalStrain(Element, IPoint3d, IPoint3d)`: retorna a direção do segmento da trinca que cortará o elemento recebido como argumento. Esta direção é definida como sendo a direção normal à máxima deformação principal não local, calculada na ponta da trinca, também recebida como argumento. Este método limita à 30 graus a variação da direção do segmento da trinca em relação ao anterior. Para tanto, recebe-se também como argumento o ponto anterior à ponta da trinca;
- + `getElemDiscontinuity()`: retorna a lista dos elementos cortados pela descontinuidade;
- + `getElements()`: retorna a lista dos elementos do modelo;
- + `getEnrichedNodes()`: retorna a lista dos nós enriquecidos devido à descontinuidade;
- + `getEnrichment()`: retorna o enriquecimento do tipo degrau associado à descontinuidade;

- + `getFirstPoint()`: retorna o primeiro ponto da descontinuidade;
- `getInitialSegmentDirectionByNonLocalPrincipalStrain(Element, IPoint3d)`: retorna a direção do segmento inicial da trinca que cortará o elemento recebido como argumento. Esta direção é definida como sendo a direção normal à máxima deformação principal não local, calculada no centroide do elemento, ponto também recebido como argumento;
- + `getLabel()`: retorna o nome da descontinuidade;
- + `getLastPoint()`: retorna o último ponto da descontinuidade;
- `getNonLocalStress(Element, IVector)`: retorna o tensor de tensões não local no ponto cujas coordenadas são passadas como argumento. Este tensor é calculado como a média ponderada das tensões nos pontos de Gauss do elemento também recebido como argumento. Para tanto, emprega-se a função de peso Gaussiana. Este método é disparado pelo método *setsCohesion* para determinação do componente de tensão na direção normal ao segmento da trinca na posição onde se introduz um ponto de Gauss relativo à coesão;
- + `setElemDiscontinuity(ArrayList<Element>)`: atribui à variável *elemDiscontinuity* a lista passada como argumento;
- + `setElements(ArrayList<Element>)`: atribui à variável *elements* a lista passada como argumento;
- + `setEnrichedNodes(ArrayList<Node>)`: atribui à variável *enrichedNodes* a lista passada como argumento;
- + `setEnrichment(DiscontinuousEnrichment)`: atribui à variável *enrichment* o objeto passado como argumento;
- + `setFirstPoint(IPoint3d)`: atribui à variável *firstPoint* o ponto passado como argumento;

- + `setLabel(String)`: atribui à variável *label* o nome passado como argumento;
- + `setLastPoint(IPoint3d)`: atribui à variável *lastPoint* o ponto passado como argumento;
- `setsCohesion(Element)`: responsável por atribuir coesão ao segmento da trinca contido no elemento passado como argumento. Para tanto, criam-se pontos de integração e definem-se os parâmetros do modelo constitutivo discreto;
- + `update()`: responsável por verificar a propagação da trinca e atualizá-la devido a esta propagação.

# Referências Bibliográficas

- Alves, P. D., 2012. Estratégia global-local aplicada ao método dos elementos finitos generalizados. Dissertação de Mestrado, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.
- Alves, P. D., Barros, F. B. e Pitangueira, R. L. S., 2013. ‘An object oriented approach to the generalized finite element method’. *Advances in Engineering Software*, vol. (59), pp. 1–18.
- Anderson, A., Zheng, X. e Cristini, V., 2005. ‘Adaptive unstructured volume remeshing - i: The method’. *Journal of Computational Physics*, vol. 208, pp. 616–625.
- Arrea, M. e Ingraffea, A., (1982), Mixed mode crack propagation in mortar and concrete., Relatório técnico 81-13, Departement Of Structural Engineering, Cornell University, Ithaca, EUA. Citado por Rots et al. (1985).
- Babuska, I., Caloz, G. e Osborn, J. E., 1994. ‘Special finite element method for a classe of second order elliptic problems whith rough coefficients.’. *SIAM Journal on Numerical Analysis*, vol. 36, pp. 3553–3576.
- Barros, F. B., 2002. Métodos Sem Malha e Métodos dos Elementos Finitos Generalizados em Análise Não-Linear de Estruturas. Tese de Doutorado, EESC - USP, São Carlos, SP, Brasil.
- Batoz, J. L. e Dhat, G., 1979. ‘Incremental displacement algorithms for nonlinear problems’. *International Journal for Numerical Methods in Engineering*, vol. 14, pp. 1262–1267.

- Boone, T., Wawrzynek, P. A. e Ingraffea, A. R., 1986. 'Simulation of the fracture process in rock with application to hydrofracturing'. *International Journal of Rock Mechanics and Mining Science*, vol. 23 (3), pp. 255–265.
- Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P. e Stal, M., 1996. *Pattern-oriented Software Architecture: A System of Patterns*. John Wiley & Sons, Inc., New York, NY, USA.
- Camarão, C., 2003. *Programação de computadores em java*. LTC.
- Carol, I., Rizzi, E. e Willam, K., 1994. 'A unified theory of elastic degradation and damage based on a loading surface'. *Int. J. Solids Structures*, vol. 31, pp. 2835–2865.
- Carol, I., Rizzi, E. e Willam, K., 2001. 'On the formulation of anisotropic elastic degradation. i. theory based on a pseudo-logarithmic damage tensor rate.'. *International Journal of Solids and Structures*, vol. 38, pp. 491–518.
- Carreira, D. J. e Chu, K., 1985. 'Stress-strain relationship for plain concrete in compression'. *ACI Journal*, vol. 82, pp. 797–804.
- Crisfield, M. A., 1981. 'A fast incremental-iterative solution procedure that handles snap-through'. *Computers & Structures*, vol. 13, pp. 55–62.
- Crisfield, M. A., 1983. 'An arc length method including line searches and accelerations'. *International Journal for Numerical Methods in Engineering*, vol. 19, pp. 1269–1289.
- Dao, L. T. N., Dao, V. T. N., Kim, S.-H. e Ann, K. Y., 2010. 'Modeling steel corrosion in concrete structures - part 2: A unified adaptive finite element model for simulation of steel corrosion'. *International Journal of Electrochemical Science*, vol. 5, pp. 314–326.
- de Borst, R. e Gutierrez, M. A., 1999. 'A unified framework for concrete damage and fracture models including size effects.'. *International Journal of Fracture* 95, pp. 261–277.

- de Vree, J. H., Brekelmans, W. A. M. e van Gils, M. A. J., 1995. ‘Comparison of nonlocal approaches in continuum damage mechanics.’. *Computers e Structures* 55 (4), pp. 581–588.
- DeJong, M. J., Hendriks, M. A. N. e Rots, J. G., 2008. ‘Sequentially linear analysis of fracture under non-proportional loading.’. *Engineering Frature Mechanics*, vol. 75 (18), pp. 5042–5056.
- Del Savio, A. A., Santi, M. R. e Martha, L. F., (2004), Traçado de curvas offset para auxilio na geração de malhas, in ‘XXV Congresso Ibero Latino Americano de Mecânica Computacional - CILAMCE, Recife, Pernambuco.’.
- Duarte, C. A. e Oden, J. T., (1995), Hp clouds - a meshless method to solve boundary-value problem., Relatório técnico, TICAM, The University of Texas at Austin.
- Duarte, C. A. e Oden, J. T., 1996. ‘An h-p adaptive method using cloud.’. *Computer Methods in Applied Mechanics and Engineering*, vol. 139, pp. 237–262.
- Dumstorff, P. e Meschke, G., 2007. ‘Crack propagation criteria in the framework of x-fem-based structural analyses’. *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 31, pp. 239–259.
- Fang, X.-J., Jin, F. e Yang, Q.-D., 2008. ‘Extended finite-element analysis of fractures in concrete’. *Engineering and Computational Mechanics*, vol. 161, pp. 187–197.
- Ferreira, G. N. A. e Pitangueira, R. L. S., (2015), Geradores de malha em uma aplicação gráfica interativa para modelos bidimensionais do método dos elementos finitos., in ‘XXXVI Congresso Ibero Latino Americano de Mecânica Computacional - CILAMCE, Rio de Janeiro, Rio de Janeiro.’.
- Ferreira, R. L., 2008. Pré-processador para modelos reticulados e planos do método dos elementos finitos. Dissertação de Mestrado, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.
- Fonseca, G. L., 1989. Geração de malhas através de mapeamentos transfinitos. Dissertação de Mestrado, PUC - Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, RJ, Brasil.



- Fuina, J. S., 2009. Formulações de Modelos Constitutivos de Microplanos para Contínuos Generalizados. Tese de Doutorado, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.
- Gamma, E., Helm, R., Johnson, R. e Vlissides, J., 1995. *Design Patterns: Elements of Reusable Object-oriented Software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
- Gonçalves, M. A. B., 2004. Geração de malhas bidimensionais de elementos finitos baseada em mapeamentos transfinitos. Dissertação de Mestrado, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.
- Goodrich, M. e Tamassia, R., 2007. *Estruturas de Dados e Algoritmos em Java - 4.ed.:*. BOOKMAN COMPANHIA ED.
- Guello, G. A., 2002. Simulação computacional de estruturas de concreto por meio da mecânica do dano. Dissertação de Mestrado, Universidade de São Paulo, São Paulo, SP, Brasil.
- Hillerborg, A., Modeer, M. e Petersson, P. E., 1976. ‘Analysis of crack formation and crack growth in concrete by means of fracture mechanics and finite elements.’. *Cement and Concrete Research*, vol. 6(6), pp. 773–782.
- Ho-Le, K., 1988. ‘Finite element mesh generation methods: a review and classification’. *Computer Aided Design*, vol. 20 (1), pp. 27–38.
- Ingraffea, A. R. e Wawrzynek, P. A., 2004. ‘Computational fracture mechanics: A survey of the field’. *European Congress on Computational Methods in Applied Sciences and Engineering ECCOMAS 2004*, pp. 1–21.
- Jirásek, M., (2001), Modeling of localized damage and fracture in quasibrittle materials, in P. A. V. et al., ed., ‘Continuous and Discontinuous Modelling of Cohesive-Frictional Materials’, Springer-Verlag, Berlin and Heidelberg, pp. 17–29.
- Jirásek, M., 2007a. ‘Mathematical analysis of strain localization’. *Damage and fracture in geomaterials*, pp. 977–991.

- Jirásek, M., (2007*b*), Mathematical analysis of strain localization, *in* ‘ALERT Doctoral School 2007 - Damage and fracture in geomaterials’, Aussois, France.
- Ju, J. W., 1989. ‘On energy-based coupled elastoplasticity damage theories: constitutive modeling and computational aspects.’. *International Journal of Solids and Structures* 25 (7), pp. 803–833.
- Karihaloo, B., 1995. *Fracture Mechanics and Structural Concrete*. Concrete design and construction series, Longman Scientific & Technical.
- Klein, P. e Gao, H., 1998. ‘Crack nucleation and growth as strain localization in a virtual-bond continuum’. *Engineering Fracture Mechanics*, vol. 61, pp. 21–48.
- Krenk, S., 1995. ‘An orthogonal residual procedure for nonlinear finite element equations’. *International Journal for Numerical Methods in Engineering*, vol. 38, pp. 823–839.
- Krenk, S. e Hededal, O., 1993. ‘A dual orthogonality procedure for nonlinear finite element equations’. *Computer Methods in Applied Mechanics and Engineering*, vol. 123, pp. 95–107.
- Lemaitre, J., 1985*a*. ‘A continuous damage mechanics model for ductile fracture.’. *Journal of Engineering Materials Technology*, vol. 107, pp. 83–89.
- Lemaitre, J., 1985*b*. ‘Coupled elasto-plasticity and damage constitutive equations.’. *Comp. Meth. Appl. Mech. Engng*, vol. 51, pp. 31–49.
- Lemaitre, J., 1992. *A Course on Damage Mechanics*. A Course on Damage Mechanics, Springer-Verlag.
- Lemaitre, J. e Chaboche, J. L., 1990. *Mechanics of Solid Materials*. Cambridge University Press, Cambridge.
- López, C. M., Carol, I. e Aguado, A., 2008. ‘Meso-structural study of concrete fracture using interface elements. i: numerical model and tensile behavior’. *Materials and Structures*, vol. 41(3), pp. 583–599.
- Mäntylä, M., 1987. *An Introduction to Solid Modeling*. Computer Science Press, Inc., New York, NY, USA.

- Mazars, J., 1984. Application de la mécanique de l'endommagement au comportement non lineaire et á la rupture du béton de structure. Tese de Doutorado, Université Paris, Paris.
- Mazars, J. e Lemaitre, J., 1984. 'Application of continuous damage mechanics to strain and fracture behavior of concrete.'. *Shah, S. P. (Ed.), Application Of Fracture Mechanics to Cementitious Composites. NATO Advanced Research Workshop, 4-7 Setembro. Northwestern University*, pp. 375–378.
- Melenk, J. M., 1995. On Generalized Finite Element Methods. Tese de Doutorado, University of Maryland, College Park.
- Mergheim, J., Kuhl, E. e Steinmann, P., 2005. 'A finite element method for the computational modelling of cohesive cracks'. *International Journal for Numerical Methods in Engineering*, vol. 63, pp. 276–289.
- Milione, Y. O., 2014. Solução de problemas estruturais dinâmicos lineares pelo método dos elementos finitos generalizado. Dissertação de Mestrado, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.
- Mohammadi, S., 2008. *Extended finite element method for fracture analysis of structures*. Blackwell Pub.
- Most, T. e Bucher, C., 2007. 'Energy-based simulation of concrete cracking using an improved mixed-mode cohesive crack model within a meshless discretization'. *International Journal For Numerical and Analytical Methods in Geomechanics*, vol. 31 (2), pp. 285–305.
- Murakami, S., 2012. *Continuum Damage Mechanics: A Continuum Mechanics Approach to the Analysis of Damage and Fracture*. Solid Mechanics and Its Applications, Springer.
- Ngo, D. e Scordelis, A. C., 1967. 'Finite element analysis of reinforced concrete beams'. *Journal of the American Concrete Institute*, vol. 64, pp. 152–163.
- Penna, S. S., 2007. Pós-processador para modelos bidimensionais não-lineares do método dos elementos finitos. Dissertação de Mestrado, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.

- Penna, S. S., 2011. Formulação Multipotencial para Modelos de Degradação Elástica - Unificação Teórica, Proposta de Novo Modelo, Implementação Computacional e Modelagem de Estruturas de Concreto. Tese de Doutorado, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.
- Pitangueira, R. L. S., 1998. Mecânica de Estruturas de Concreto com Inclusão de Efeitos de Tamanho e Heterogeneidade. Tese de Doutorado, PUC - Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro.
- Pituba, J. J. C., 1998. Estudo e aplicação de modelos constitutivos para o concreto fundamentados na mecânica do dano contínuo. Dissertação de Mestrado, Universidade de São Paulo, São Carlos, SP, Brasil.
- Planas, J., Guinea, G. V. e Elices, M., 1999. 'Size effect and inverse analysis in concrete fracture.'. *International Journal of Fracture*, vol. 95, pp. 367–378.
- Ramm, E., 1981. 'Strategies for tracing the nonlinear response near limit points, in nonlinear finite element analysis in structural mechanics'. , pp. 63–83.
- Ricks, E., 1972. 'The application of newton method to the problem of elastic stability'. *Journal of Applied Mechanics*, pp. 1060–1065.
- Ricks, E., 1979. 'An incremental approach to the solution of snapping and buckling problems'. *International Journal of Solids and Structures*, vol. 15, pp. 529–551.
- Rots, J. G., 1988. Computational modeling of concrete fracture. Tese de Doutorado, Delft University of Technology, Delft, Holanda.
- Rots, J. G., Nauta, P., Kusters, G. M. e Blaauwendrra, J., 1985. 'Smearred crack approach and fracture localization in concrete'. *HERON 30*, vol. (1), pp. 1–48.
- Shah, S. P., Swartz, S. E. e Ouyang, C., 1995. *Fracture Mechanics of Concrete: Applications of Fracture Mechanics to Concrete, Rock and other Quasi-Brittle Materials*. John Wiley & Sons, Inc., 605 Third Avenue, New York, NY 10158-0012.
- Simo, J. C. e Ju, J. W., 1987. 'Strain- and stress-based continuum damage models - i. formulation.'. *International Journal of Solids and Structures 23 (7)*, pp. 821–840.

- Süssekind, J. C., 1981. *Curso de Análise Estrutural I: Estruturas Isostáticas*. Vol. 1 (6), Editora Globo S. A.
- Thiagarajan, G., Hsia, K. J. e Huang, Y., 2004. 'Finite element implementation of virtual internal bond model for simulating crack behavior'. *Engineering Fracture Mechanics*, vol. 71, pp. 401–423.
- Wells, G. N. e Sluys, L. J., 2001. 'A new method for modelling cohesive cracks using finite elements'. *International Journal for Numerical Methods in Engineering*, vol. 50(12), pp. 2667–2682.
- Winkler, B., Hofstetter, G. e Lehar, H., 2004. 'Application of a constitutive model for concrete to the analysis of a precast segmental tunnel lining'. *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 28, pp. 797–819.
- Wolenski, A. R. V., Monteiro, A. B., Barros, F. B., Pitangueira, R. L. S. e Penna, S. S., (2014), Application of the generalized finite element method on the physically nonlinear analysis of structures., in 'XXXV Congresso Ibero Latino Americano de Mecânica Computacional - CILAMCE, Fortaleza, Ceará.'
- Wolff, K. P., 2010. Implementação computacional de um modelo de fissuração para o concreto baseado no método dos elementos finitos estendido (xfem). Dissertação de Mestrado, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil.
- Yang, Y. B. e McGuire, W., 1985. 'A work control method for geometrically nonlinear analysis'. *Proceedings of the International Conference on Numerical Methods in Engineering: Theory and Application*, pp. 913–921.
- Yang, Y. B. e Shieh, M. S., 1990. 'Solution method for nonlinear problems with multiple critical points'. *AIAA Journal*, vol. 28(12), pp. 2110–2116.
- Zheng, X., Lowengrub, J., Anderson, A. e Cristini, V., 2005. 'Adaptive unstructured volume remeshing - ii: Application to two- and three-dimensional level-set simulations of multiphase flow'. *Journal of Computational Physics*, vol. 208, pp. 626–650.