3
Trabalhos Anteriores

Neste capítulo faremos uma revisão das principais estruturas de dados topológicas existentes na literatura. Classificaremos os trabalhos de acordo com a classe de objetos que estes podem representar. Na seção 3.1 citaremos estruturas para superfícies em $\mathbb{R}^3$, na seção 3.2 estruturas desenvolvidas para a representação de variedades de dimensão 3, na seção 3.3 estruturas para variedades de dimensão n e, por fim, em 3.4 estruturas para não–variedades.

Alguns trabalhos citados neste capítulo serão descritos com mais detalhe, já que, no capítulo 6, os compararemos com as estruturas CHE e CHF.

PUC-Rio - Certificação Digital Nº 0410406/CA

3.1
Estruturas de Dados para Superfícies em $\mathbb{R}^3$

A estrutura *Winged–Edge* de (Baumgart 1975) foi um dos primeiros trabalhos propostos para a representação de superfícies em $\mathbb{R}^3$. Muitas modificações desta estrutura foram desenvolvidas com o objetivo de aumentar a variedade de objetos a serem modelados. Entre tais modificações, os trabalhos mais importantes são o de (Braid *et al.* 1980), que introduziu o conceito de *loop*, e o de (Mäntylä 1988), que criou a estrutura de dados *Half–Edge*.

Em seguida, (Guibas and Stolfi 1985), com a *Quad–Edge*, também generalizaram a *Winged–Edge*, permitindo a modelagem de variedades de dimensão 2 não orientáveis. Ainda, (Castelo *et al.* 1992) (Lopes 1996) definiram a *Handle–Edge*: uma estrutura de dados que representa explicitamente o bordo da superfície combinatória, o que permite uma descrição direta da conectividade de suas células de bordo.

Mais recentemente, (Rossignac *et al.* 2001) propuseram uma estrutura de dados extremamente concisa chamada *Corner–Table* que utiliza apenas vetores de inteiros e um conjunto de regras para representar superfícies trianguladas.

A primeira estrutura de dados que usou o conceito de escalabilidade, foi batizada como *Directed–Edges*, e proposta por (Campagna *et al.* 1998).

Escolhemos as estruturas *Handle–Edge*, *Corner–Table* e *Directed–Edges* para serem descritas com mais detalhes, já que em cada delas observamos características semelhantes às características da CHE.

Handle–Edge A *Handle–Edge* é uma extensão da estrutura de dados *Half–Edge* desenvolvida com o objetivo de se obter uma implementação eficiente dos operadores de alça (Lopes 1996). Para tal a *Handle–Edge* propõe uma representação explícita das curvas de bordo da superfície. A estrutura trabalha com o conceito de *half–edge*, definido por (Mäntylä 1988). Uma *half–edge* é uma aresta orientada em um triângulo da malha. A implementação originalmente proposta para a *Handle–Edge* utiliza ponteiros, e cada entidade é representada por um tipo de nó. A seguir descreveremos cada nó da estrutura de dados:

- **Nó Surface:** O *nó surface* representa a instância de uma superfície. Este nó é conectado aos primeiros elementos das listas duplamente encadeadas de vértices, faces, arestas e curvas de bordo que compõem a superfície (figura 3.1).

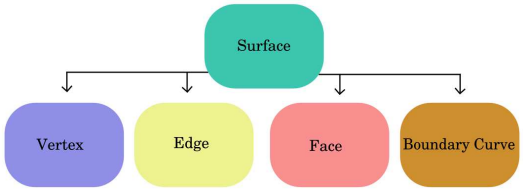

Figura 3.1: Nó surface.

- **Nó Face:** O *nó face* representa um triângulo (que também chamaremos de *face*) da superfície. Toda face é delimitada por um ciclo de *half–edges*. Cada face é ligada a uma de suas *half–edges* incidentes e à superfície da qual faz parte. As faces são elementos de uma lista composta por todas as faces da superfície (figura 3.2).

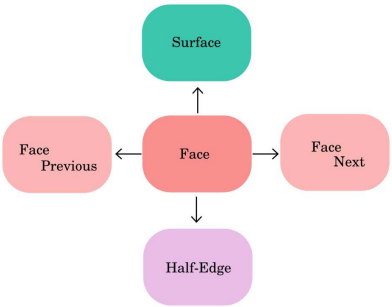

Figura 3.2: Nó face

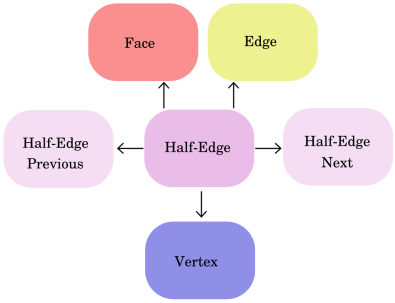

Figura 3.3: Nó half–edge

- **Nó Half–Edge:** Um *nó half–edge* representa, explicitamente, a orientação de uma aresta na face a que pertence, induzida a partir da orientação da face (figuras 3.4, 3.5). Cada nó *half–edge* é ligado à face incidente e ao vértice inicial da *half–edge*. Ainda, guardamos referências

PUC-Rio - Certificação Digital Nº 0410406/CA

da aresta incidente e das *half–edges* seguinte e anterior (chamadas de *next* e *previous*) no ciclo de *half–edges* da face incidente (figura 3.3).

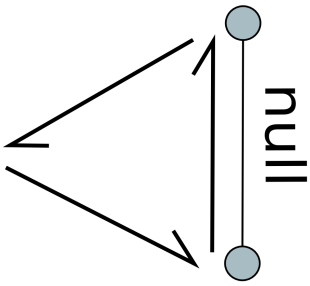

Figura 3.4: Half–edge de uma aresta de bordo.

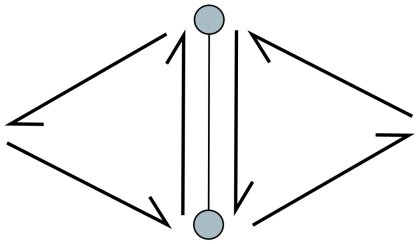

Figura 3.5: Half–edges de uma aresta de interior.

- **Nó Boundary Curve:** Um *nó boundary curve* representa uma componente conexa do bordo da superfície através de uma de suas arestas incidentes (figura 3.6). O nó *boundary curve* faz parte de uma lista composta por todas as curvas de bordo da superfície.

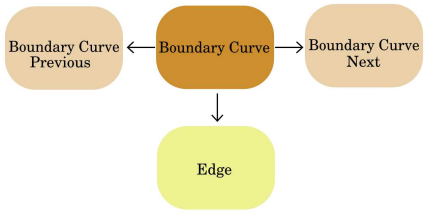

Figura 3.6: Nó Boundary Curve.

- **Nó Edge:** Um *nó edge* pode representar tanto uma aresta de bordo quanto uma aresta de interior (figura 3.7). Cada aresta é ligada a um trio ordenado $(HE_1, HE_2, \partial S)$ composto de dois nós *half–edge* e um nó *boundary curve*, e construído de acordo com as seguintes regras:
  - Se a aresta é de interior, o trio tem a configuração $(HE_1, HE_2, \emptyset)$, onde HE_1 e HE_2 são as duas *half–edges* incidentes a aresta.
  - Se a aresta é de bordo, a configuração do trio é $(HE_1, \emptyset, \partial S)$, onde HE_1 é sua única *half–edge* e ∂S é a curva de bordo a que pertence a aresta.

  O nó edge também faz parte de uma lista composta por todas as arestas de uma superfície.

- **Nó Vertex:** O *nó vertex* pode representar tanto vértices de bordo quanto vértices de interior (figura 3.8). Cada vértice é associado a um par ordenado de *half–edges* que obedece as seguintes regras:

PUC-Rio - Certificação Digital Nº 0410406/CA

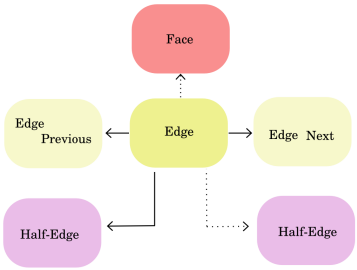

Figura 3.7: Nó edge

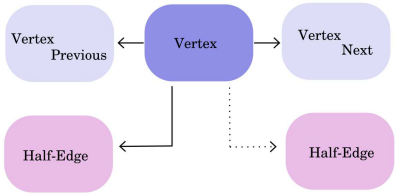

Figura 3.8: Nó vertex

– Se o vértice for de interior, a primeira *half–edge* armazena é uma das *half–edges* incidentes ao vértice, e a segunda aponta para *null* (figura 3.9).
– Se o vértice é um vértice de bordo, as duas *half–edges* de bordo incidentes são armazenadas. A primeira corresponde a aresta que tem o vértice como ponto inicial, e a segunda a aresta que tem o vértice como ponto final (figura 3.10).

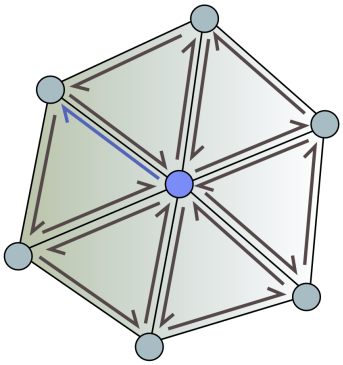

Figura 3.9: Vértice de interior.

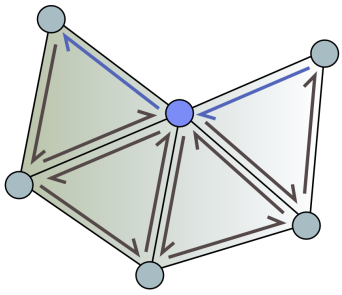

Figura 3.10: Vértice de bordo.

Corner–Table A *Corner–Table* é uma estrutura de dados concisa para a representação de malhas triangulares que trabalha com um conceito equivalente ao de *half–edge*: o corner. Um *corner* é a associação de um triângulo a um de seus vértices, enquanto uma *half–edge* é a associação de um triângulo e uma de suas arestas. As duas definições são equivalentes pois, para cada *corner* em um triângulo, sempre existe uma única *half–edge* oposta a ele (figura 3.11).

Cada triângulo é representado por três *corners* consecutivos, indexados segundo sua orientação. Logo, uma malha com $\mathbf{n}_2$ triângulos terá $3\mathbf{n}_2$ *corners* e, por definição, o triângulo i será representado pelos *corners* $3i$, $3i+1$, $3i+2$.

Desse modo, podemos escrever expressões simples para o cálculo dos

PUC-Rio - Certificação Digital Nº 0410406/CA

corners "vizinhos" a um *corner* de índice c no triângulo $\lfloor c/3 \rfloor$ [1], chamados de *next* de c e *previous* de c (denotados por $next(c)$ e $prev(c)$) [2]:

$$\begin{aligned} next(c) &:= 3 * \lfloor c/3 \rfloor + (c+1)\%3, \\ prev(c) &:= 3 * \lfloor c/3 \rfloor + (c+2)\%3. \end{aligned}$$

Os atributos geométricos da *Corner–Table* são guardados em um vetor chamado G[]. Para cada vértice da malha, associamos atributos como coordenadas, vetores normal etc. A dimensão de G[] é igual a $\mathbf{n}_0$.

As informações topológicas carregadas pela estrutura, ficam armazenadas em dois vetores de inteiros batizados por V[] e O[], ambos de dimensão $3\mathbf{n}_2$. O vetor V[] armazena o índice do vértice associado a um determinado corner. Para construir o vetor O[] temos que definir o conceito de *corner* oposto. Diremos que dois *corners* de índices c e o são *opostos* se $\mathsf{V}[next(c)] = \mathsf{V}[prev(o)]$ e $\mathsf{V}[prev(c)] = \mathsf{V}[next(o)]$ (figura 3.12). O vetor O[] guarda o *corner* oposto de cada corner.

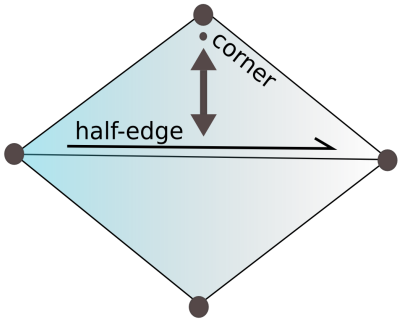

Figura 3.11: Relação corner/half–edge.

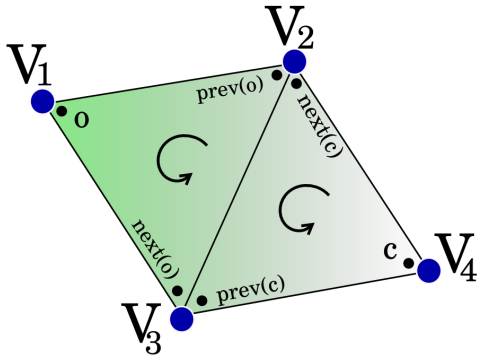

Figura 3.12: Corners opostos c e o.

Apenas com as tabelas V[] e O[] e as duas regras aritméticas descritas é possível atravessar eficientemente os triângulos de uma malha representada pela estrutura. Assim a vizinhança de um vértice pode ser obtida a partir de um de seus *corner* incidente. Porém, a localização de um *corner* a partir do índice de um vértice tem custo linear já que é preciso atravessar a tabela V[] a sua procura. Uma solução simples para este problema, proposta por (Vieira 2003), é armazenar, para cada vértice, o índice de um de seus *corners* incidentes.

Directed–Edges A Estrutura de dados *Directed–Edges* é uma extensão escalonável da *Half–Edge*. Existem 3 níveis chamados de *full–sized*, *medium–sized* e *small–sized*. A estrutura é baseada no conceito de *directed–edge*. Uma

[1] O símbolo $\lfloor * \rfloor$ representa a parte inteira do valor de $*$. Ou seja, um *corner* c pertence ao triângulo de índice igual a parte inteira da divisão de c por 3.

[2] O símbolo $a\%b$ nas expressões dos *corners next* e *previous* representa o resto da divisão de a por b. Neste caso estamos trabalhando com o resto da divisão por 3.

PUC-Rio - Certificação Digital Nº 0410406/CA

directed–edge e^d de uma aresta e é uma ocorrência de e no bordo de um de seus triângulos incidentes. Vale observar, que o conceito de *directed–edge* é também equivalente aos conceitos de *half–edge* e *corner* utilizados pelas estruturas apresentadas anteriormente (figura 3.13).

Na estrutura *Directed–Edges*, triângulos e arestas não são explicitamente representados, entretanto são implicitamente relacionados com suas *directed–edges* incidentes: o triângulo com índice i, denotado por f^i, é descrito pelas *directed–edges* $3i$, $3i+1$ e $3i+2$, que formam o bordo orientado de f^i, já uma aresta geométrica é descrita pelo par de *directed–edges* incidentes.

Para cada *directed–edge*, podemos armazenar referências para seus vértices, que chamamos de v^a e v^b, para sua *directed-edge* vizinha, isto é aquela incidente à mesma aresta geométrica, denotada por e^{ng}, e para as outras *directed–edges* no bordo do triângulo incidente, identificadas por e^{pv} e e^{nx} (figura 3.14). A memória utilizada para armazenar algumas dessas referências pode ser substituída por uma quantidade constante de cálculos adicionais.

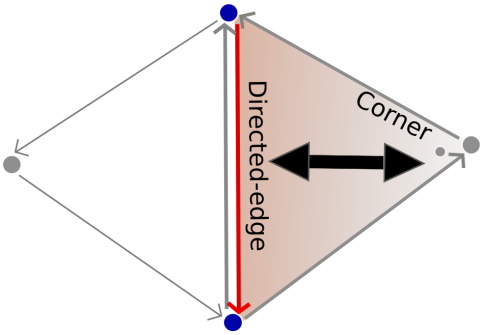

Figura 3.13: Relação corner/directed–edge.

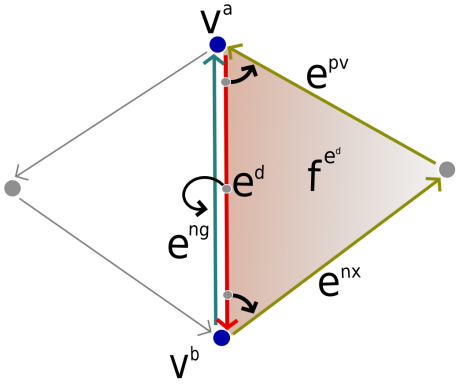

Figura 3.14: Informações armazenadas pela directed–Edge e^d.

Os níveis da estrutura codificam, para cada *directed–edge*:

- *Small–Sized*: O vértice final v^b e a *directed–edge* vizinha e^{ng}.
- *Medium–Sized*: O vértice final v^b, a *directed–edge* vizinha e^{ng} e a *directed–edge* anterior e^{pv}, seguindo a orientação do triângulo incidente f^i.
- *Full–Sized*: Os vértices inicial e final v^a e v^b, a *directed–edge* vizinha e^{ng} e as *directed–edges* anterior e posterior e^{nx} e e^{pv}, seguindo a orientação do triângulo incidente f^i.

Para otimizar a estrutura podemos armazenar, para cada vértice, uma de suas *directed–edges* incidentes. Ainda, podemos representar arestas e vértices não variedade armazenando uma *directed–edge* incidente para cada componente de bordo da estrela do simplexo não variedade.

PUC-Rio - Certificação Digital Nº 0410406/CA

3.2 Estruturas de Dados para 3–Variedades

Existem várias estruturas de dados desenvolvidas para a representação do domínio de uma 3–variedades através de uma decomposição simplicial. A abordagem descrita em (Guibas and Stolfi 1985) foi estendida por (Dobkin and Laszlo 1989), através de uma nova proposta chamada *Facet–Edge*, que tinha como objetivo representar complexos celulares que são subdivisões de uma esfera tridimensional. Posteriormente, o conceito de *half–edge* foi estendido por (Lopes and Tavares 1997), com a estrutura de dados *Handle–Face*, que representava explicitamente a superfície de bordo de 3–variedades, permitindo o controle de sua topologia.

A seguir, estudaremos com mais detalhes a estrutura de dados *Handle–Face* pois esta apresenta características bastante parecidas com as da CHF.

Handle–Face A *Handle–Face* é uma extensão da *Handle–Edge* para variedades de dimensão 3 que utiliza o conceito de *half–face*. Uma *half–face* é um 2–simplexo dotado de uma orientação induzida por um de seus 3–simplexos incidentes. A implementação da *Handle–Face* utiliza ponteiros e cada entidade é representada por um nó.

O grande diferencial da estrutura é a introdução do *nó boundary surface*, que nos dá uma representação explícita para o bordo da 3–variedade. Quando a superfície de bordo é explicitamente representada, o acesso às relações de incidências e adjacências nas faces de bordo, arestas de bordo e vértices de bordo é otimizado, permitindo uma implementação eficiente dos operadores de alça.

- **Nó 3–Manifold:** Um *nó 3–manifold* é representado por uma lista de superfícies (que são bordo de volumes), uma lista de superfícies de bordo, uma lista de faces de interior, uma lista de aresta de interior, e por fim uma lista de vértices de interior (figura 3.15).

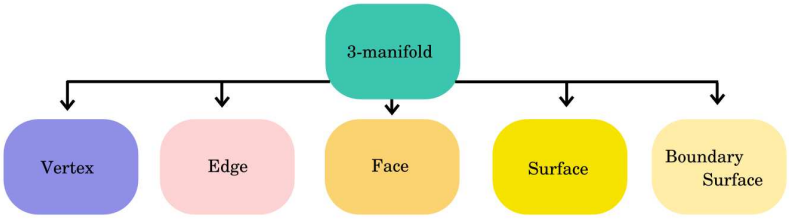

Figura 3.15: Nó 3–Manifold.

PUC-Rio - Certificação Digital Nº 0410406/CA

- **Nó Surface:** Um *nó surface* corresponde a um sólido da variedade (tetraedro). Nele guardamos referências para a lista de *half–faces*, a lista de arestas e a lista de vértices do sólido. Ainda o nó aponta para a 3–variedade a que pertence o sólido (figura 3.16).
- **Nó Boundary Surface:** Um *nó boundary surface* corresponde a uma componente conexa da superfície de bordo da 3–variedade. Cada superfície é representada pelos seus simplexos incidentes. São armazenadas referências para as listas de faces, arestas e vértices da componente de bordo. O nó aponta para a 3–variedade a que pertence (figura 3.17).

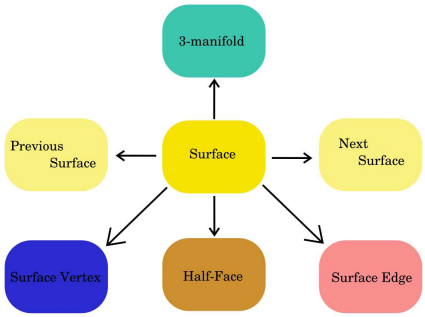

Figura 3.16: Nó Surface.

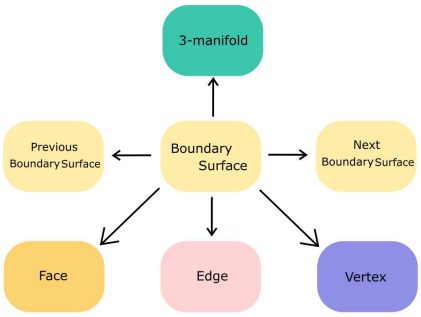

Figura 3.17: Nó Boundary Surface.

- **Nó Face:** Um *nó face* pode representar tanto uma face de interior quanto uma face de bordo (figura 3.18).

  No caso de uma face de interior, o nó é conectado às suas duas *half–faces* incidentes. Caso contrário, o nó representa uma face de bordo e é conectado a sua única *half–face* e a superfície de bordo que contém a face.

- **Nó Half–Face:** Um *nó half–face* representa uma *half–face* de uma célula de dimensão 3, através de seu ciclo de *half–edges*. Ainda, ele armazena informações sobre a face e o sólido incidentes à *half–face* (figura 3.19).

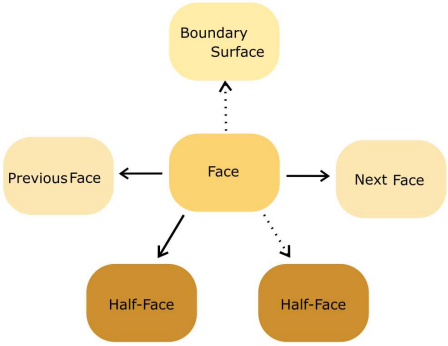

Figura 3.18: Nó Face.

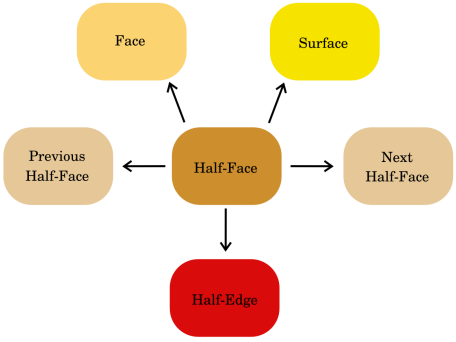

Figura 3.19: Nó Half–Face.

PUC-Rio - Certificação Digital Nº 0410406/CA

- **Nó Edge:** Um *nó edge* pode corresponder a uma aresta de interior ou a uma aresta de bordo (figura 3.20).

  Se a aresta for de bordo, o nó tem uma representação direta das duas *half–edges* de bordo correspondentes a esta aresta. Por outro lado, se a aresta for interior à variedade, armazenamos apenas uma de suas *half–edges* incidentes para que sejamos capazes de descobrir o ciclo de *half–edges* em torno de uma aresta.

- **Nó Surface Edge:** O *nó surface edge* representa uma aresta na superfície de uma célula de dimensão 3. Cada aresta superfície tem duas *half–edges* incidentes e uma referência para a aresta correspondente da 3–variedade. Ainda, armazenamos as *half–edges* oposta e radial de cada *half–edges* incidentes (figura 3.21).

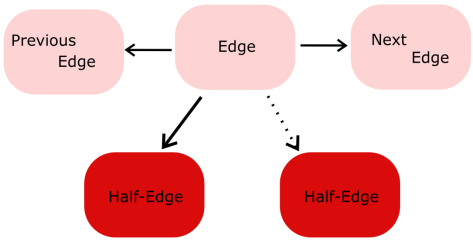

Figura 3.20: Nó Edge.

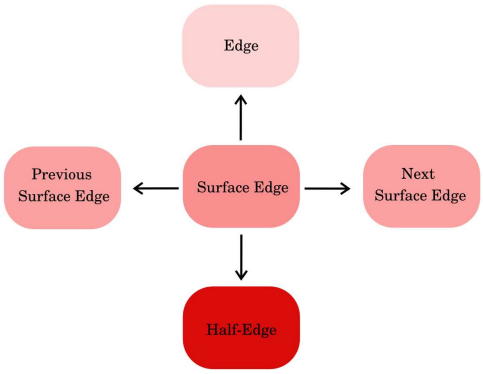

Figura 3.21: Nó Surface Edge.

- **Nó Half–Edge:** Um *nó half–edge* armazena referências diretas para as *half–edges next* e *previous*, para o vértice inicial, para a *half–face* incidente e para a aresta incidente de uma *half–edge* (figura 3.22).

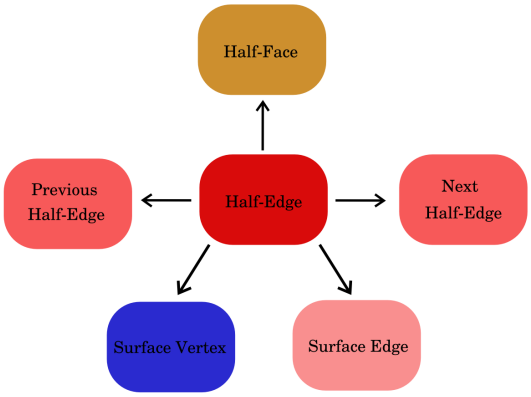

Figura 3.22: Nó Half–Edge.

PUC-Rio - Certificação Digital Nº 0410406/CA

- **Nó Vertex:** O *nó vertex* representa um vértice da variedade e armazena uma lista com todos os vértices superfície incidentes a ele. Se o vértice estiver no bordo da 3–variedade armazenamos também uma de suas *half–edges* de bordo incidentes (aquela cujo ponto inicial é o vértice) (figura 3.23).

- **Nó Surface Vertex:** O *nó surface vertex* tem uma ligação direta com o vértice original da 3–variedade. Ainda é armazenada uma *half–edge* da superfície que tem o vértice como ponto inicial para que seja possível o cálculo eficiente da estrela na superfície correspondente (figura 3.24).

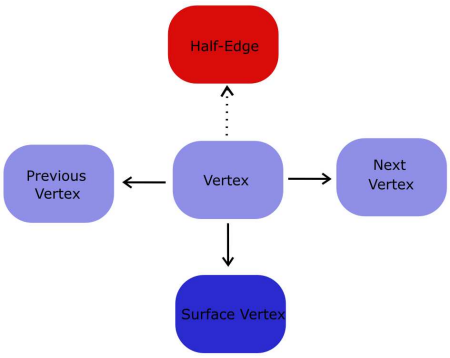

Figura 3.23: Nó Vertex.

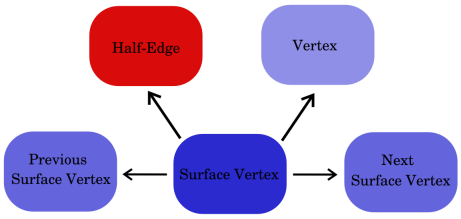

Figura 3.24: Nó Surface Vertex.

A figura 3.25 representa uma visão global do relacionamento entre os nós da *Handle–Face*.

3.3 Estruturas de Dados para n–Variedades

De acordo com (Brisson 1993), dentre todas as estruturas de dados citadas anteriormente, apenas a *Quad–Edge* e a *Facet–Edge* não representam explicitamente cada célula da estrutura combinatória. Por outro lado, a *Winged–Edge*, a *Half–Face* e a *Handle–Edge* são exemplos de estruturas de dados que fazem tal representação explícita das células.

Generalizando a idéia da *Quad–edge* e da *Facet–Edge*, alguns trabalhos significativos foram introduzidos com o objetivo de representar n–variedades. Entre eles, as estruturas *Cell–Tuple* de (Brisson 1993), *the n–generalized Maps for simplicial quasi–manifolds* de (Lienhardt 1994) e *Hypermaps* de (Bertrand and Dufourd 1994) merecem ser citadas.

Um outro exemplo de estrutura, independente da dimensão, é a *Indexed data structure with adjacencies* proposta por (Paoluzzi *et al.* 1993).

Apresentaremos também com mais detalhes a estrutura de dados *Indexed data structure with adjacencies* pois, suas características bastante parecidas com as da CHF.

PUC-Rio - Certificação Digital Nº 0410406/CA

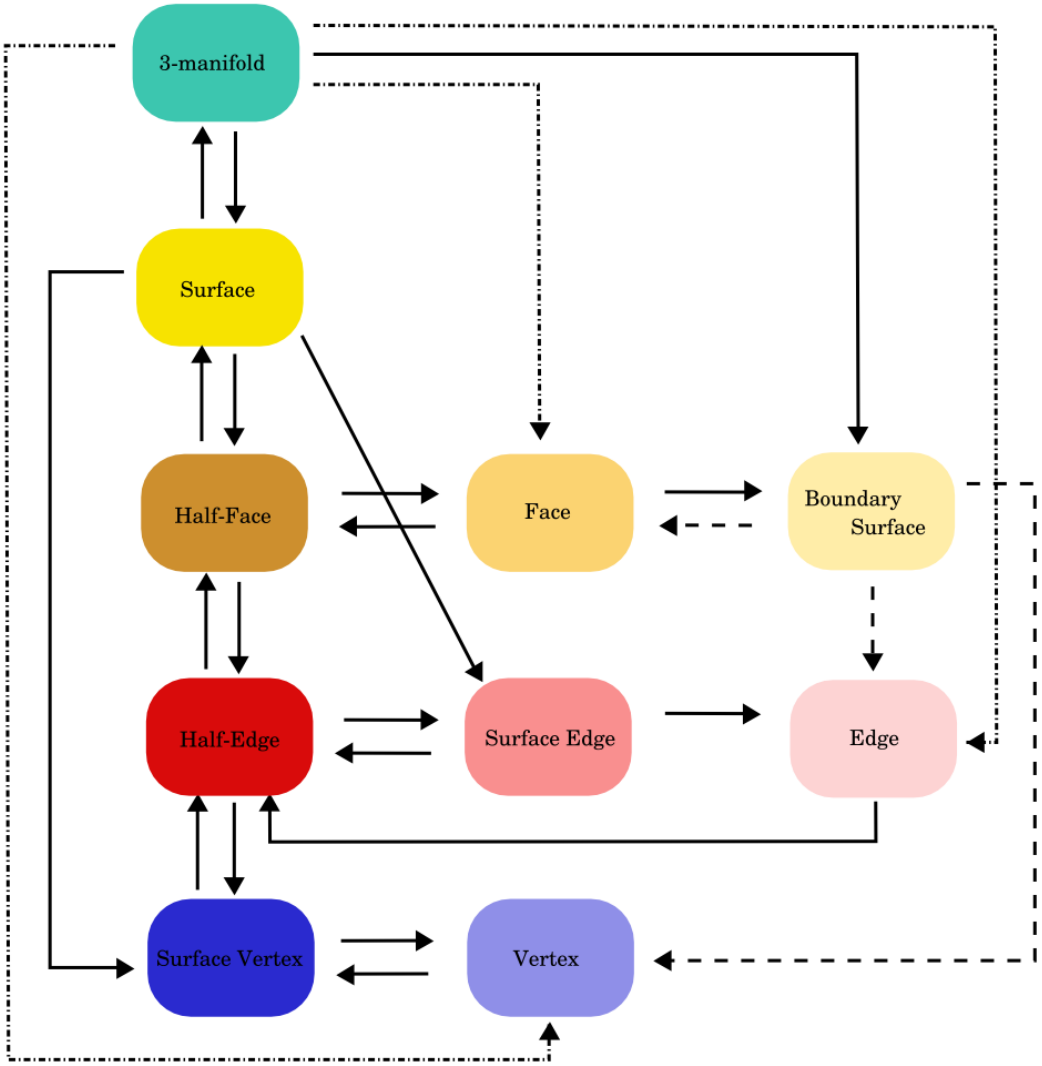

Figura 3.25: Nós da Handle–Face.

Indexed Data Structure with Adjacencies A *indexed data–structure with adjacencies*, ou simplesmente *Ia*, é uma estrutura de dados compacta para representação de variedades de dimensão arbitrária. A estrutura representa apenas o conjunto dos simplexos de dimensão máxima no complexo simplicial, e as relações de adjacência entre tais simplexos.

Na implementação, a *Ia* utiliza um vetor de tamanho $\mathbf{n}_0$ para a representação das coordenadas de cada vértice, e um par de vetores, cada um de tamanho $d \cdot \mathbf{n}_{d\text{-}1}$, para a armazenar os vértices e relação topológica entre as $(d-1)$–faces dos d–simplexos.

Observamos facilmente que as relações de bordo podem ser recuperadas eficientemente, entretanto a estrutura de dados não suporta navegação eficiente pelas arestas e vértices da malha.

PUC-Rio - Certificação Digital Nº 0410406/CA

3.4 Estruturas de Dados para não–Variedades

A modelagem de não–variedades é fundamental em muitas aplicações. Por exemplo, em sistemas de modelagem, precisamos ter uma representação eficiente de não–variedades já que o conjunto das variedades não é fechado para operadores booleanos.

A primeira proposta de estrutura de dados para este tipo de objeto foi feita por (Weiler 1986), com a estrutura *Radial–Edge*. Em seguida, várias alterações da estrutura foram desenvolvidas com objetivo de adaptá-la a novas aplicações. Algumas contribuições substanciais para a representação de não–variedades são os trabalhos de (Wu 1989, Wu 1992, Yamaguchi and Kimura 1995, Gursoz *et al.* 1990, Rossignac and O'Connor 1990, Cavalcanti *et al.* 1997, Lee and Lee 2001).

Ainda, (Pesco *et al.* 2003) propuseram a estrutura *Handle–Cell*, uma extensão da *Handle–Face*, que faz o tratamento de 2–complexos celulares gerais. Por fim, (Nonato *et al.* 2001) propuseram outra extensão para a *Handle–Face* para tratar casos especiais de singularidade, a *Singular Handle–Face*.

Mais recentemente (Floriani and Hui 2003) apresentaram uma estrutura bastante concisa para a manipulação de não–variedades chamada *NMIA*, que é uma extensão da *Indexed data structure with adjacencies*.

Apresentaremos também com mais detalhes a estrutura de dados *NMIA* pois, no trato de 3–variedades, suas características bastante parecidas com as da CHF.

NMIA A *Non–Manifold Indexed data structure with adjacencies* (*NMIA*) é uma estrutura de dados para a representação de não–variedades de dimensão 3. Sua proposta é representar os simplexos topo de dimensões 1 e 2, chamados de *wire–edges* e *dangling–faces* (figuras 3.26 e 3.27). Mais ainda, é preciso que a estrutura possa descrever as estrelas de um vértice ou de uma aresta nos casos em que estas contêm mais de uma componente conexa. Resumindo, podemos agrupar as singularidades tratadas pela estrutura em quatro casos:

1. Existência de vértices cuja estrela é formada por mais de uma componente conexa, chamados de *nm–vértice* (figura 3.26).
2. Existência de *wire–edges* (figura 3.26).
3. Existência de arestas cuja estrela é formada por mais de uma componente conexa, chamadas de *nm–aresta* (figura 3.27).
4. Existência de *dangling–faces* (figuras 3.26 e 3.27).

Para trabalhar com os casos de singularidade listados, um novo grupo de *funções de resposta* precisa ser definido: as relações $R_{*clusters}$. A função $R_{0clusters}$

PUC-Rio - Certificação Digital Nº 0410406/CA

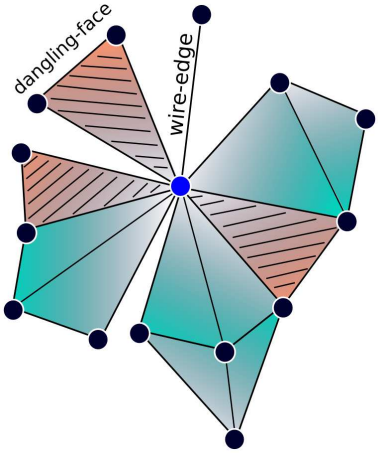

Figura 3.26: Estrela de um *nm–* vértice não-variedade

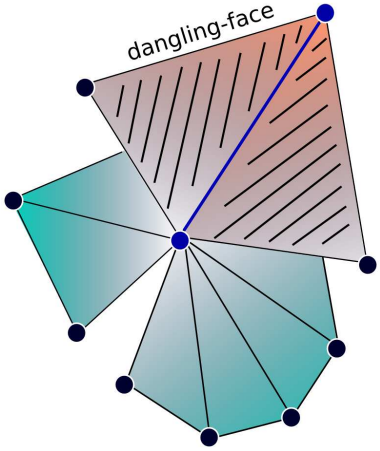

Figura 3.27: Estrela de uma *nm–* aresta não-variedade

retorna, para um vértice v, o conjunto de simplexos topo formado pelos representantes de cada componente conexa da estrela de v. A função $R_{2clusters}$ retorna, para cada aresta e_i de uma *dangling–face*, o grupo de simplexos topo formado pelos representantes de cada componente conexa da estrela de e_i, ordenados no sentido anti–horário. A função $R_{3clusters}$ é análoga a $R_{2clusters}$ para as arestas de um tetraedro .

As informações topológicas armazenadas pela estrutura de dados são:

- Para cada *tetraedro*: as relações R_{30}, R_{33} e $R_{3clusters}$.
- Para cada *dangling–face*: a relação R_{20} e $R_{2clusters}$.
- Para cada *wire–edge*: a relação R_{10}.
- Para cada *vértice*: a relação $R_{0clusters}$.

Assim, dado um complexo simplicial Σ a *NMIA* codifica todos os vértices, *wire–edges*, *dangling–faces* e tetraedros de Σ. A estrutura de dados não representa explicitamente as arestas e faces do complexo simplicial.

PUC-Rio - Certificação Digital Nº 0410406/CA