4
Técnicas de Filtragens Aplicadas às Visões do Ambiente de Autoria do Sistema HyperProp

Um problema enfrentado pelos usuários que trabalham com estruturas de dados grandes é a desorientação na busca por determinada informação. Essa desorientação é causada pelo número elevado de vértices e arestas.

Dentre as técnicas mais comuns para a apresentação das estruturas, podem ser citadas o uso de recursos de *zoom* e *scroll* em partes da estrutura. Entretanto, a técnica que se mostra mais eficaz, embora mais complexa, é a filtragem de partes da estrutura não relevantes para o usuário. A dificuldade é identificar quais partes da estrutura realmente são relevantes. Uma solução muito aplicada nos editores de estruturas complexas para filtrar informações é a técnica olho-de-peixe (Furnas, 1986).

Este capítulo está dividido da seguinte forma: na Seção 4.1 será explicada a técnica de filtragem olho-de-peixe e sua extensão para grafos compostos. Para um melhor entendimento da técnica olho-de-peixe utilizada no sistema HyperProp, a Seção 4.2 está dividida em três subseções. A Subseção 4.2.1 apresenta a técnica olho-de-peixe sobre a visão estrutural do sistema HyperProp com base no exemplo ilustrado na Seção 4.1. A Subseção 4.2.2 demonstra o funcionamento da visão olho-de-peixe na visão espacial na especificação de *layout* sincronizada com a visão declarativa. Por fim, a Subseção 4.2.3 analisa os cálculos realizados para filtragem na especificação de *layout* na visão espacial.

4.1.
Técnica de Filtragem Olho-de-Peixe aplicada em Grafos Compostos

A visão olho-de-peixe, proposta por Furnas (Furnas, 1986) para estruturas hierárquicas, atua como uma lente, preservando os detalhes próximos a um ponto escolhido (vértice em foco) e, à medida que se afasta desse ponto, exibindo menos informação (apenas as informações mais importantes segundo critérios que serão explicados no próximo parágrafo).

A estratégia do olho-de-peixe é definir uma função de grau de interesse, que atribui a cada elemento do grafo um valor que representa o grau de interesse do usuário em relação ao elemento em foco. A idéia principal é que essa função grau de interesse (*DOI(x,y)*) aumente com a importância *a priori* (*API(x)*) do elemento *"x"* e diminua com a distância (*D(x,y)*) entre os elementos "*x*" e "*y*", sendo o elemento *"y"* o foco. Assim, tem-se: ***DOI(x,y) = API(x) – D(x,y).***

A filtragem dos vértices é realizada escolhendo um determinado valor *"k"* para o corte, de modo a exibir somente os elementos *"x"* que possuírem um $DOI(x,y) \geq k$. Variando o valor de *"k"*, que pode ser interpretado como o nível de detalhe desejado, obtém-se diferentes visões olho-de-peixe para o mesmo grafo.

A proposta original de Furnas é aplicada em estruturas onde a função *DOI(x,y)* pode ser facilmente definida, como em listas e árvores. Para a utilização da técnica olho-de-peixe no sistema Hyperprop, a mesma teve de ser estendida para ser aplicada em grafos compostos (Muchaluat-Saade et al., 1998), levando em conta tanto as relações de aninhamento como relações de arestas entre os vértices para calcular quais os vértices e arestas são visíveis.

No caso de grafos compostos, a importância *a priori API(x)* de um vértice *"x"* é dada pelo valor negativo do nível de aninhamento do vértice observado em relação ao vértice composto mais externo (no caso, o próprio grafo ou um vértice composto do grafo que tenha sido eleito como o escopo da filtragem[1]). Assim, ***API(x) = -i*** onde " ***i*** " é o nível de profundidade do vértice em relação ao vértice composto eleito como escopo. A Figura 4-1 ilustra um exemplo de cálculo da função *API(x)* para grafos compostos.

[1] Se um vértice composto é eleito como sendo o escopo para filtragem significa que apenas os vértices diretamente ou recursivamente contidos no vértice composto podem vir a ser exibidos. No caso *default*, o próprio grafo pode ser entendido como o vértice composto eleito como escopo.

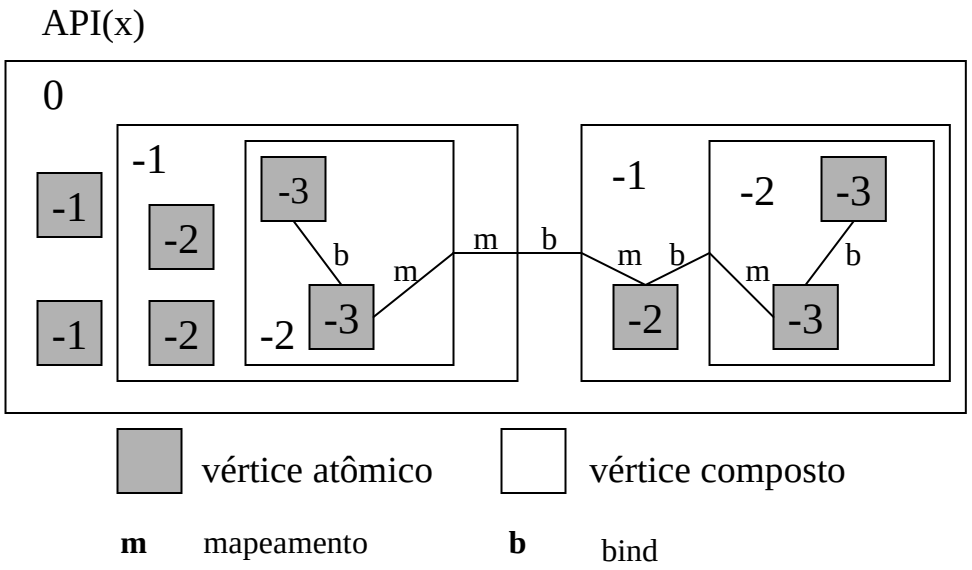

Figura 4-1 Exemplo de cálculo de API(x) para grafos compostos

A segunda componente da função *DOI*, a distância *D(x,y)*, é computada entre um vértice *"x"* e o vértice *"y"* (vértice em foco) da seguinte forma:

D(x,y) = min(Dc(x,y) + wc, De(x,y) + we).

A função *Dc(x,y)* é a distância mínima entre os vértices *"x"* e *"y"* considerando somente a relação de aninhamento das estruturas de composição, que será explicada melhor nos próximos parágrafos, e a função *De(x,y)* é a distância mínima entre *"x"* e *"y"* considerando somente os relacionamentos por arestas. Se não há um caminho de arestas ligando os dois vértices, a distância *D(x,y)* é equivalente a *Dc(x,y)*. As constantes *"wc"* e *"we"* representam os pesos que estes dois métodos de navegação (*Dc(x,y)* e *De(x,y)*) têm no cálculo da visão olho-de-peixe. A diferença entre esses pesos determina uma prioridade na exibição dos vértices relacionados ao vértice em foco pela estrutura de composição ou por arestas.

A distância *Dc(x,y)*, considerando-se a navegação em profundidade (baseada apenas no aninhamento de vértices compostos), do foco *"y"* para o vértice *"x"* é dada por: $\boldsymbol{Dc(x,y) = dist_a + dist_d}$.

O valor de $\boldsymbol{dist_a}$ **é a distância ascendente**, calculada adicionando-se uma unidade para cada passo (passagem de um vértice para seu respectivo pai (vértice composto)) no caminho ascendente desde o vértice "y" (foco) até que o primeiro vértice composto ancestral comum aos vértices *"x"* e *"y"* seja alcançado.

O valor de $dist_d$ é a distância descendente, calculada adicionando-se uma unidade para cada passo (passagem para o vértice filho (composto ou atômico)) no caminho descendente entre o vértice composto ancestral comum encontrado no cálculo da *distância* $dist_a$ e *"y"*. Ao final, se a distância do caminho descendente é maior que zero, soma-se o valor calculado com o valor de $dist_a$. Em outras palavras, ***$dist_d$ = caminho descendente + $dist_a$*** **, se caminho descendente > 0**. O valor $dist_a$ é usado no cálculo de $dist_d$ para manter a relação de distância do vértice ancestral comum e o vértice com foco. A Figura 4-2 ilustra um exemplo de cálculo da função *Dc(x,y)* para grafos compostos.

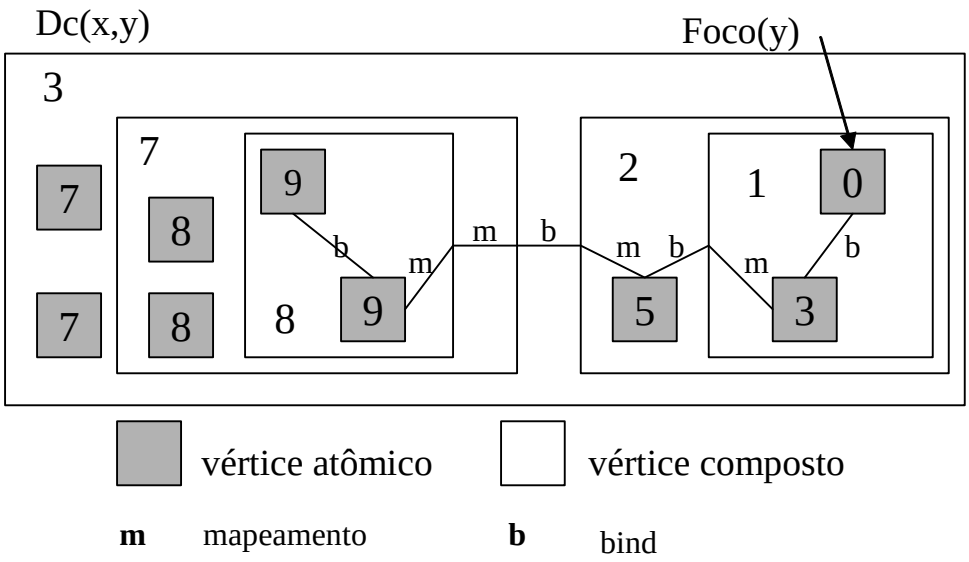

Figura 4-2 Exemplo de cálculo de *Dc(x,y)* para grafos compostos

Para calcular a distância considerando a navegação por arestas *De(x,y)*, deve-se, para cada aresta (*binds* e mapeamentos, vide Seção 2.1) percorrida entre os vértices *"x"* e *"y"*, adicionar uma unidade.

A distância *De(x,y)* será igual ao valor mínimo encontrado entre todos os caminhos possíveis entre *"x"* e *"y"*. Note que pode existir mais de um caminho relacionando os vértices *"x"* e *"y"* por arestas. Para tal cálculo de *De(x,y)*, pode ser usado o algoritmo de Dijkstra (Cormen et al., 2001), que calcula o menor caminho entre dois vértices num grafo não-orientado. A Figura 4-3 ilustra um exemplo de cálculo da função *De(x,y)* para grafos compostos.

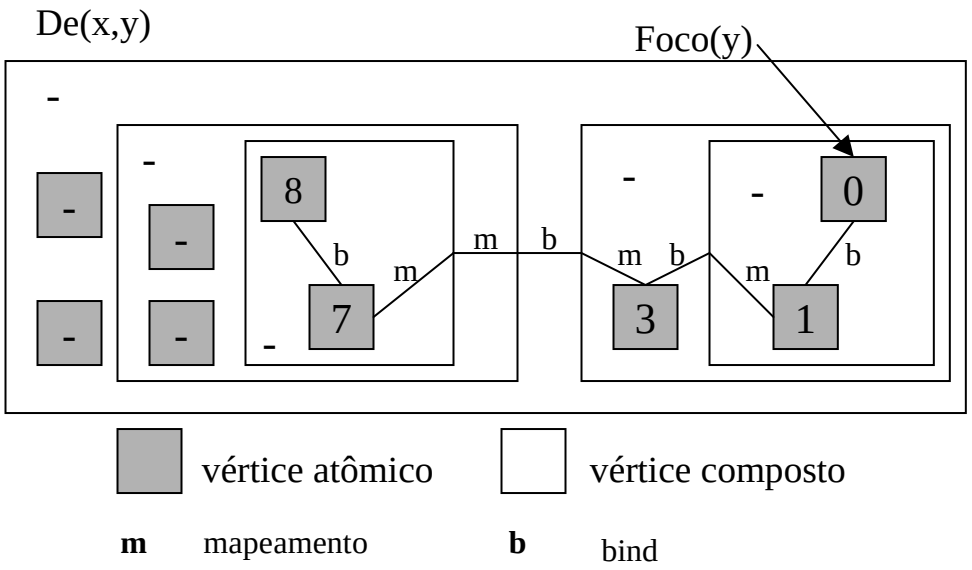

Figura 4-3 Exemplo de cálculo de *De(x,y)* para grafos compostos

Apenas a título de ilustração, a Figura 4-4 apresenta os valores dos vértices considerando a função *D(x,y) = min(Dc(x,y) + wc , De(x,y) + we)* e as distâncias das Figuras 4-2 (*Dc(x,y)*) e 4-3 (*De(x,y*)). Para esse caso em particular, as constantes *wc* e *we* foram atribuídas com valor zero.

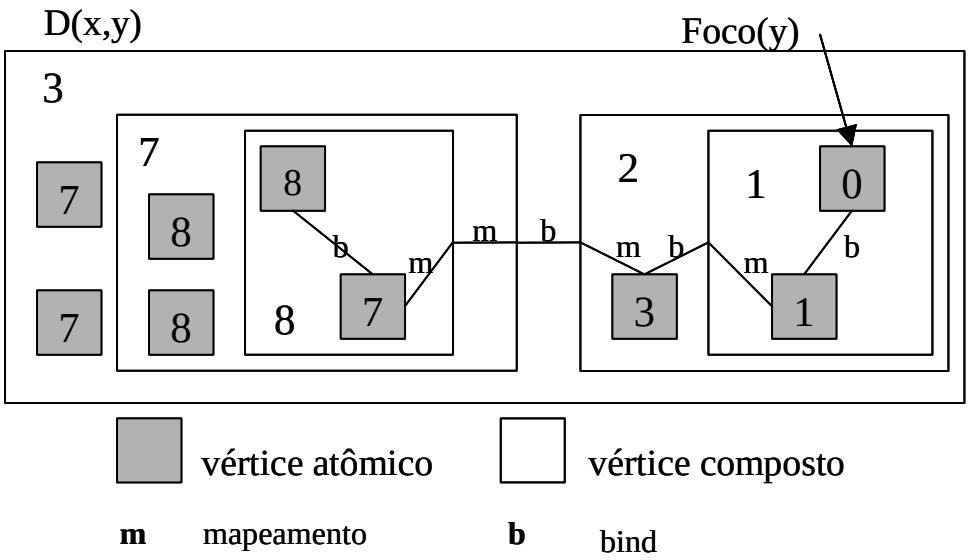

Figura 4-4 Exemplo de cálculo da função *D(x,y)* para grafos compostos baseado nas Figuras 4-2 (*Dc(x,y)*) e 4-3 (*De(x,y)*)

A Figura 4-5 apresenta os valores dos vértices para a função *DOI(x,y) = API(x) – D(x,y)* baseados na Figuras 4-1 (*API(x)*) e 4-4 (*D(x,y)*).

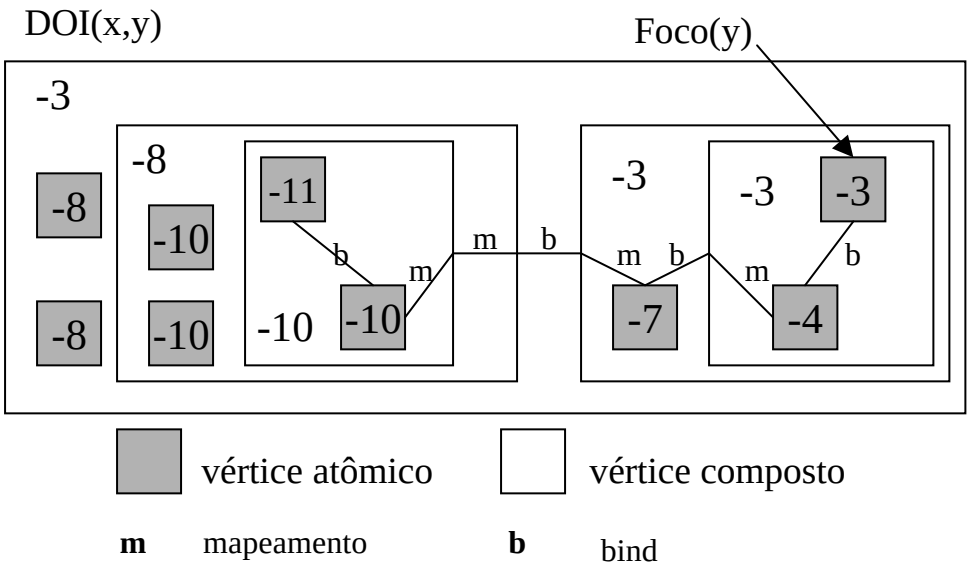

Figura 4-5 Exemplo de cálculo da função *DOI(x,y)* para grafos compostos baseado nas Figuras 4-1 (*API(x)*) e 4-4 (*D(x,y)*)

Caso o usuário escolha o valor de *"k"* igual a menos cinco, por exemplo, somente os vértices com grau de interesse maior ou igual a menos cinco serão visualizados pelo usuário.

4.2. Filtragem Olho-de-Peixe no Sistema HyperProp

O sistema HyperProp, na implementação atual, utiliza a filtragem olho-de-peixe nas visões estrutural, declarativa e na especificação do leiaute na visão espacial, podendo a mesma ser aplicada a qualquer momento durante a edição e navegação dos documentos. Conforme o usuário altera o parâmetro *"k"* ou seleciona um novo vértice como foco, os demais vértices são exibidos e/ou excluídos nas visões espacial, estrutural e declarativa, de forma sincronizada.

4.2.1.
Filtragem olho-de-peixe na Visão Estrutural

A técnica de filtragem olho-de-peixe foi reintegrada à visão estrutural do sistema HyperProp e sincronizada com a visão declarativa. Para usar a visão olho-de-peixe na visão estrutural, o usuário deve inicialmente ativar a técnica de filtragem clicando na opção *"View"* da barra de *menu* da visão estrutural e, em seguida, selecionar na opção *"FishEye"*. Nesse instante, todos os vértices compostos são fechados na visão do grafo.

A Figura 4-6 ilustra a visão estrutural do sistema HyperProp apresentando um grafo composto com todos seus vértices desenhados. Nesse momento, a técnica de filtragem olho-de-peixe ainda não foi aplicada sobre o grafo.

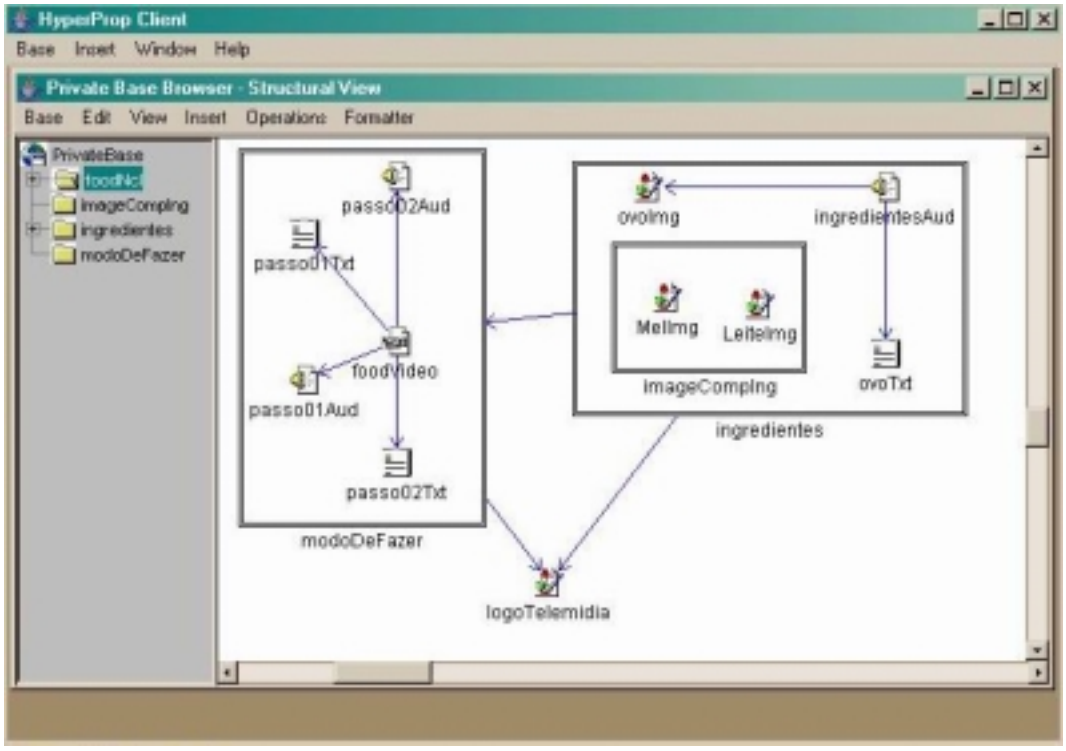

Figura 4-6 Visão estrutural sem a técnica olho-de-peixe

A escolha do vértice a ser utilizado como foco é realizada após a ativação da filtragem. Para definir um vértice como foco, o usuário deve selecioná-lo com o botão direito do *mouse* e, em seguida, escolher a opção *"Focus"* entre as alternativas de *menu* exibida sobre o vértice. Os parâmetros de configuração (*k*, *wc* e *we*) da visão olho-de-peixe podem ser alterados através da janela *"FishEye Configuration"*.

Para exibir a janela *"FishEye Configuration"* o usuário deve selecionar a opção *"View"* da barra de *menu* da visão estrutural e depois selecionar a alternativa *"Configure"*.

Observe na Figura 4-7 que em um determinado momento da navegação o usuário escolheu como foco o vértice "*melImg*". Em seguida, ativou a janela *"FishEye Configuration"* para configurar a visão olho-de-peixe com nível de detalhe no valor de "*25%*" de sua visualização inicial. Isto fez com que apenas os vértices *melImg*, *imageCompImg* e *ingredientes* fossem exibidos.

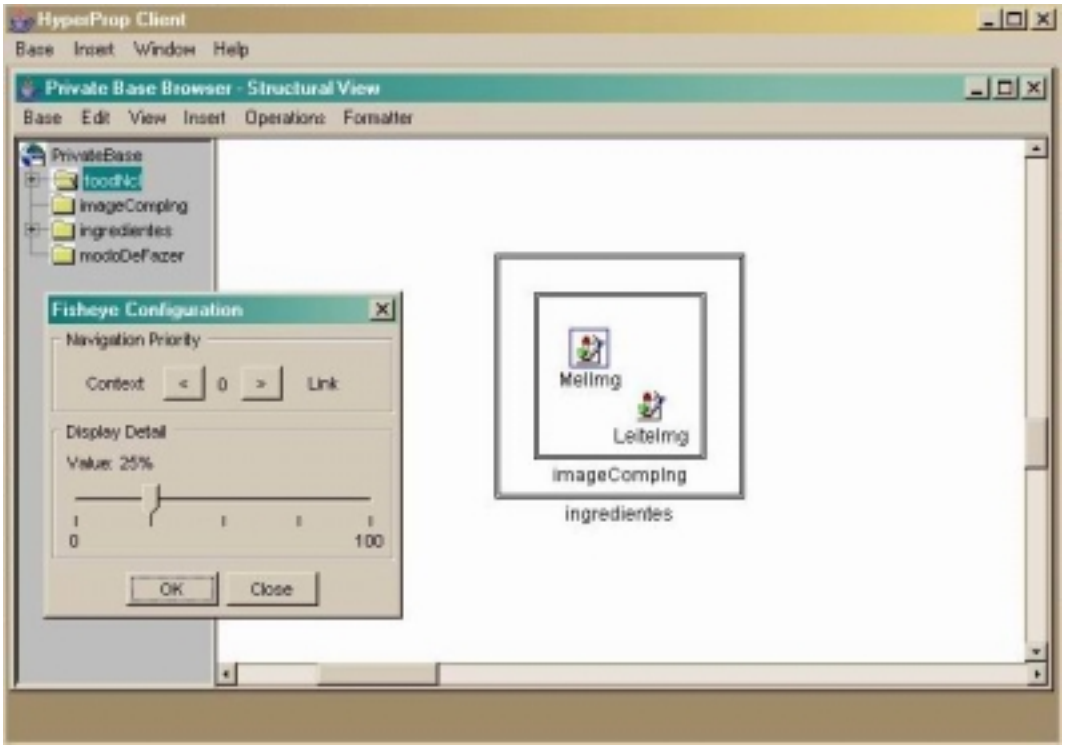

Figura 4-7 Visão olho-de-peixe com 25% de *Nível de detalhe*

Quando o usuário ativa visão olho-de-peixe, os vértices exibidos na visão estrutural são automaticamente apresentados na visão declarativa (na parte direita do editor) com tamanhos variados de texto, diferenciando-os de acordo com a respectiva distância ao foco.

Já a parte esquerda do editor declarativo apresenta apenas os vértices exibidos na visão estrutural. A Figura 4-8 mostra a visão declarativa sincronizada com a visão estrutural.

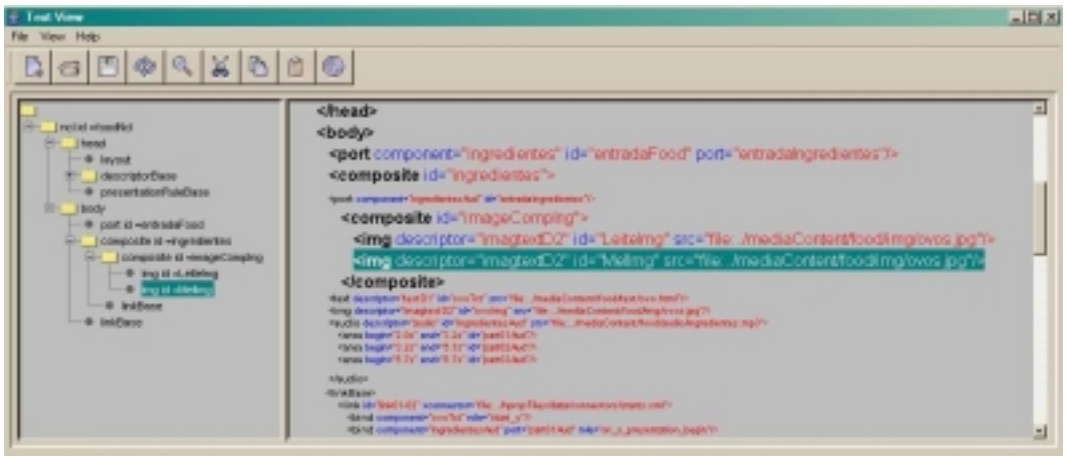

Figura 4-8 Visão declarativa sincronizada com a visão estrutural (olho-de-peixe 25%)

Conforme o usuário navega no grafo ou altera os parâmetros de configuração da visão olho-de-peixe na janela *"FishEye Configuration"*, as novas informações exibidas na visão estrutural são refletidas na visão declarativa de forma sincronizada.

A Figura 4-9 ilustra o grafo na visão estrutural com *nível de detalhe* no valor de "*75%*" de visualização e a Figura 4-10 mostra a visão declarativa sincronizada com a visão estrutural da Figura 4-9. Observe a diferença de tamanho nas fontes de texto na parte direita do editor.

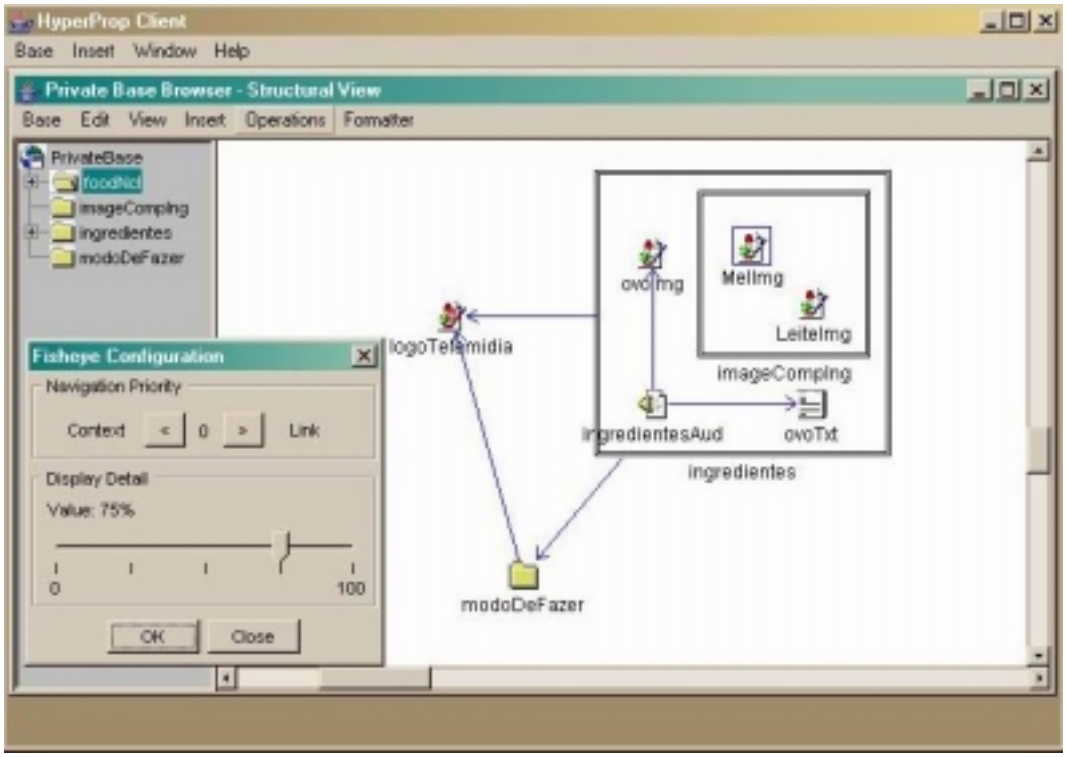

Figura 4-9 Visão olho-de-peixe com 75% de *Nível de detalhe*

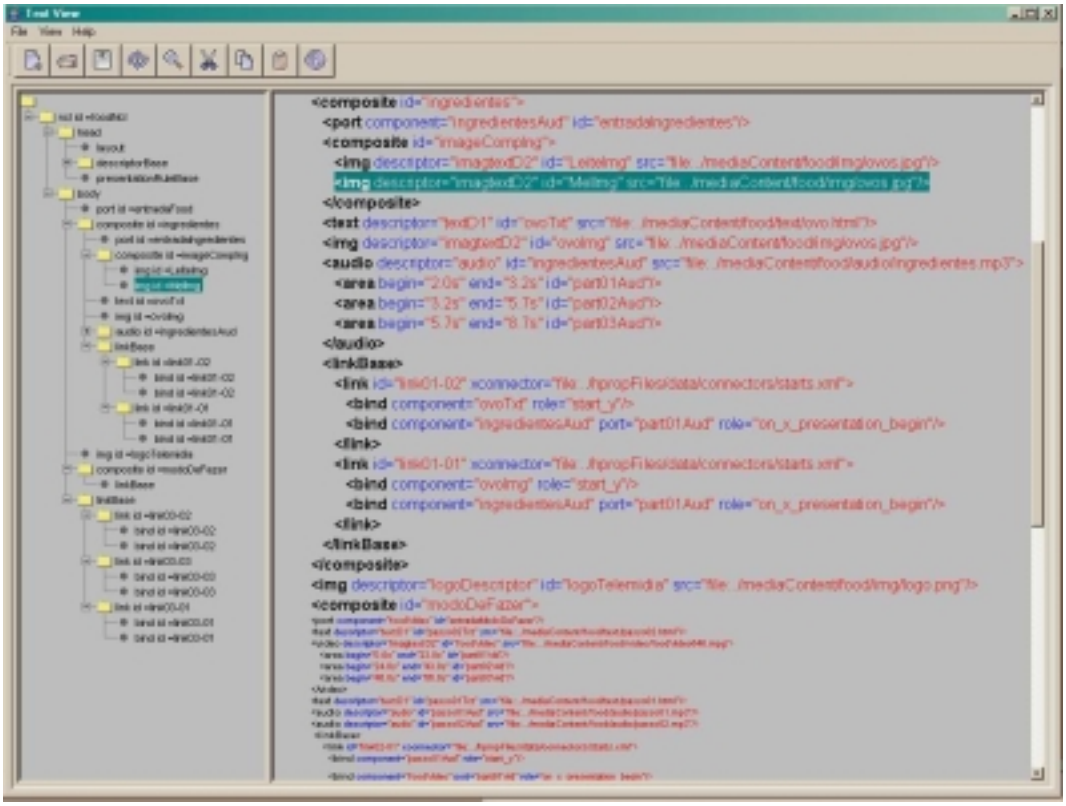

Figura 4-10 Visão declarativa sincronizada com visão estrutural (olho-de-peixe 75%)

4.2.2.
Visão olho-de-peixe na Visão Espacial e Textual

Conforme já apresentado na Seção 3.4, durante a edição espacial, o usuário pode especificar o leiaute da apresentação espacial de uma determinada arquitetura de acordo com os recursos disponíveis.

A Figura 4-11 ilustra o editor espacial na especificação de um leiaute para apresentação de documentos hipermídia baseado na linguagem NCL. Nessa figura, o autor definiu três elementos *topLayout* (*"w1","w2" e "w3"*) com vários filhos internos a eles (*regions).* Além disso, escolheu como foco o elemento *region "r8"*. A mesma informação apresentada na visão espacial da Figura 4-11 pode ser visualizada na visão textual conforme a Figura 4-12.

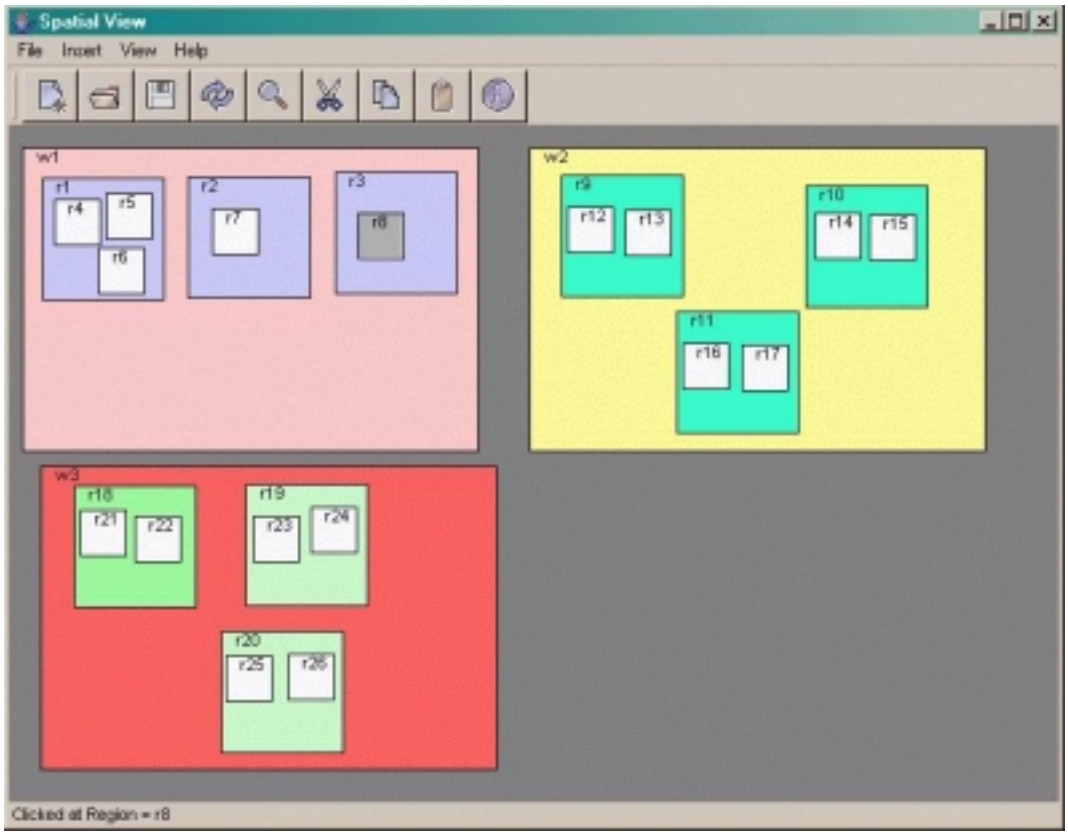

Figura 4-11 Visão Espacial da especificação do elemento *layout* na linguagem NCL

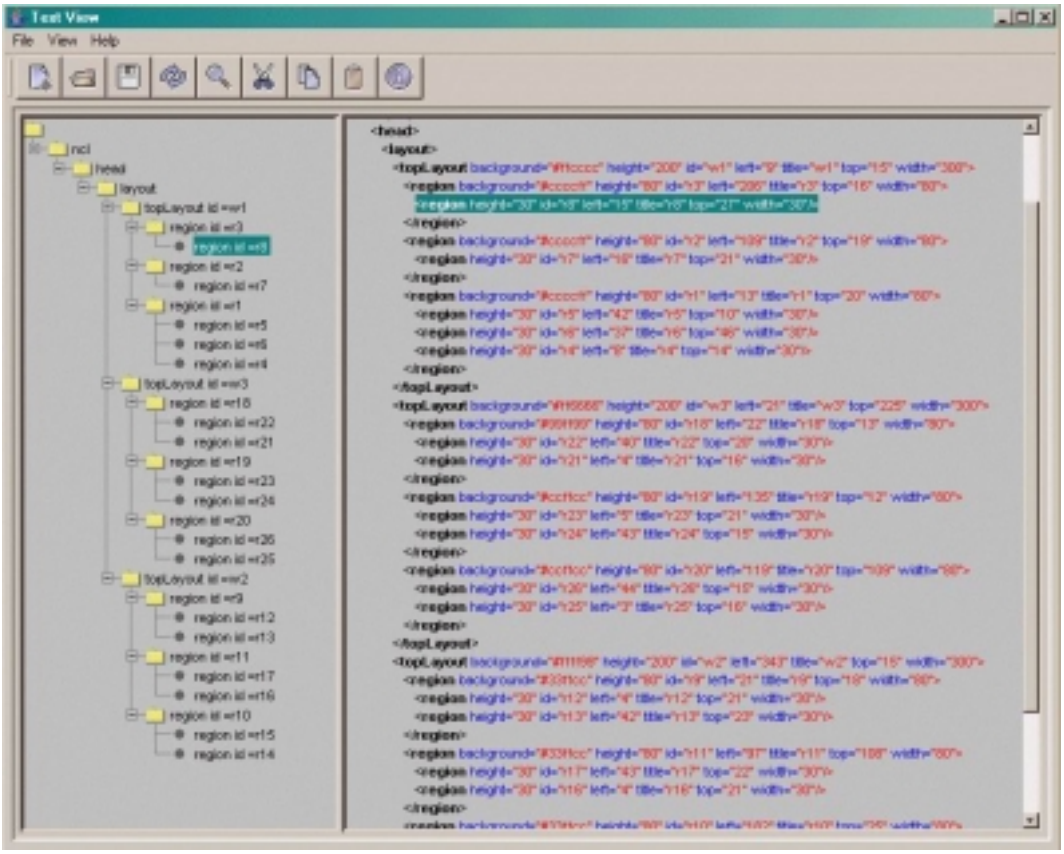

Figura 4-12 Visão Textual da especificação do elemento *layout* refletida da Figura 4-11

Para aplicar a técnica olho-de-peixe na edição do leiaute na visão espacial, o usuário inicialmente seleciona um vértice como foco e, em seguida, ajusta o parâmetro *nível de detalhe* (parâmetro *"k"*) na janela *"FishEye Configuration"*. O filtro aplicado na visão espacial é refletido na visão textual. A Figura 4-13 ilustra a visão espacial com a filtragem olho-de-peixe mantendo a região *"r8"* como foco e usando um *nível de detalhe* em aproximadamente 50%.

Conforme o usuário altera o *nível de detalhe* (parâmetro *"k"*) ou seleciona um novo vértice como foco, a filtragem é automaticamente reavaliada e a visão atualizada.

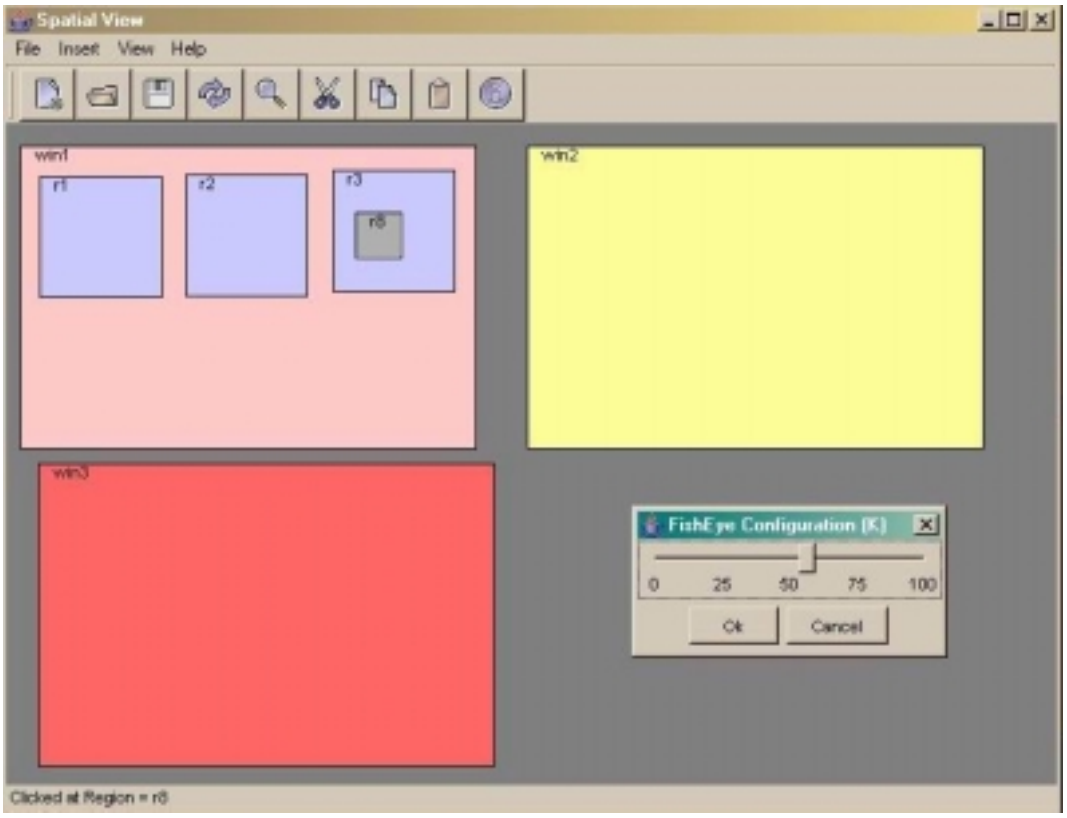

Figura 4-13 Visão olho-de-peixe aplicada na visão espacial

A Figura 4-14 ilustra a visão textual sincronizada com a visão olho-de-peixe aplicada sobre o editor espacial da Figura 4-13. Observe que, na visão textual, os vértices não ilustrados no editor espacial da Figura 4-13 são excluídos da parte esquerda do editor (visão em árvore), enquanto que, na área direita (texto), os elementos são mostrados com tamanhos variados de texto, diferenciando-os de acordo com a respectiva distância ao foco. Quanto mais próximo do foco o vértice estiver, maior será o tamanho do texto exibido no editor.

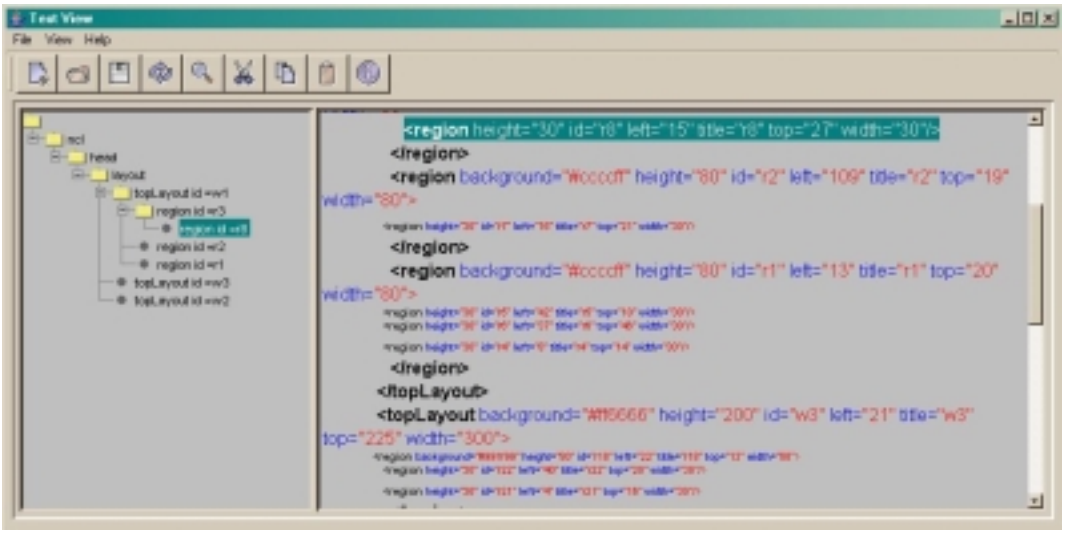

Figura 4-14 Visão olho-de-peixe aplicada na visão textual

4.3. Cálculos Realizados na Filtragem Olho-de-Peixe sobre a Visão Espacial

Conforme mencionado na Seção 4.1, a técnica olho-de-peixe pode ser aplicada na forma original proposta por Furnas (Furnas, 1986) para estruturas simples como **árvores** e listas.

Os relacionamentos de inclusão entre os elementos presentes na linguagem NCL (*layout*, *topLayout* e *region*) visualizados na Figura 4-12 podem ser tratados como uma **árvore**, onde a raiz da árvore (vértice composto) é o elemento *layout* da linguagem NCL.

É importante destacar que o algoritmo olho-de-peixe aplicado sobre a especificação de **um leiaute na visão espacial** foi aplicado na forma **original** proposta por Furnas, diferente da **visão estrutural**, em que o algoritmo foi adaptado para grafos compostos conforme discutido na Seção 4.1.

O cálculo realizado na especificação do leiaute da visão espacial da Figura 4-12 (foco sobre o vértice "*r8*") procedeu nas seguintes etapas:

1. calcular os valores da função API(x) - no caso da estrutura em árvore, cada elemento pertencente à árvore recebe um valor **negativo** referente a seu nível de profundidade (**API(x) = - i**). A Figura 4-15 ilustra os elementos da visão espacial com seus respectivos valores calculados;

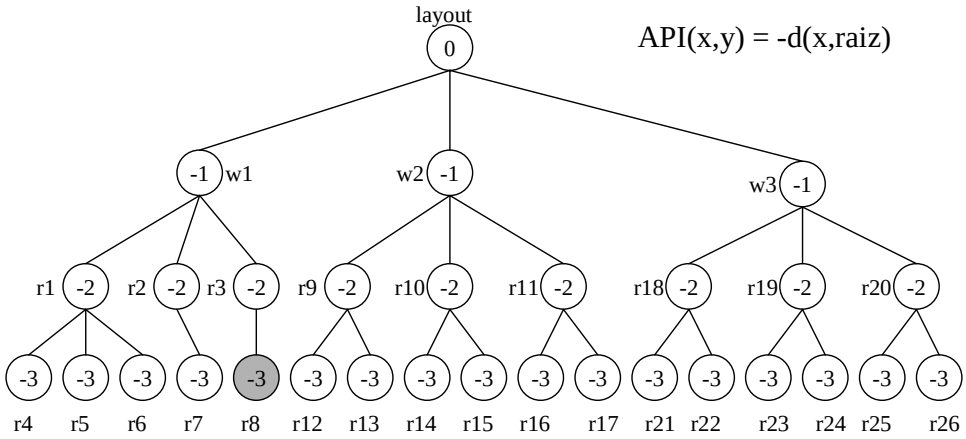

Figura 4-15 Cálculo da função API(x,y)

2. calcular os valores da função D(x,y) - no caso da distância do vértice *"x"* em relação ao vértice *"y"* (foco), soma-se uma unidade para cada aresta entre o vértice em foco (*"r8"*) e o vértice de destino. A Figura 4-16 ilustra os elementos com seus respectivos valores calculados;

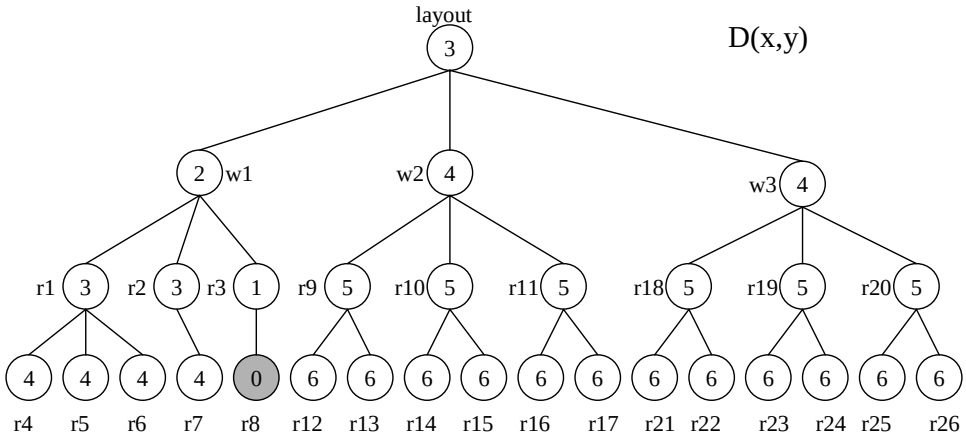

Figura 4-16 Cálculo da função D(x,y)

3. calcular os valores da função DOI(x,y). Para estruturas em árvores foi utilizado *DOI(x,y) = API(x) – D(x,y)*. A Figura 4-17 ilustra os elementos com seus respectivos valores calculados;

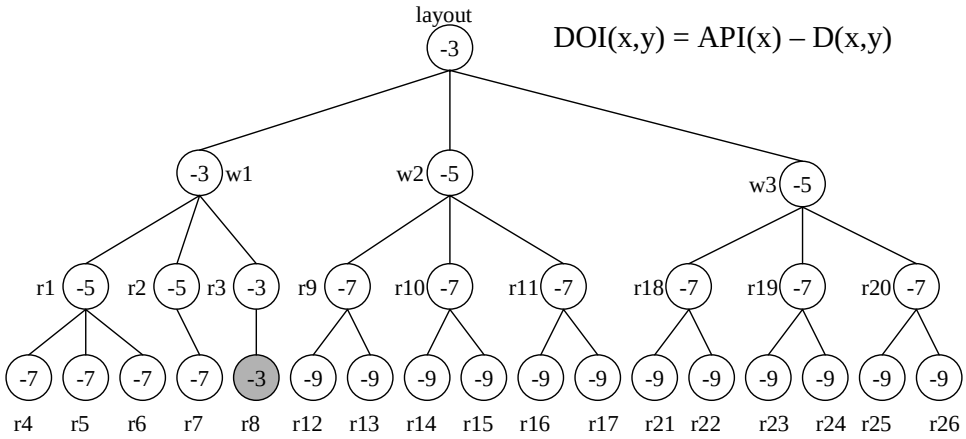

Figura 4-17 Cálculo da função DOI(x,y)

4. selecionar quais elementos devem ser exibidos: conforme o usuário altera o *nível de detalhe* um novo parâmetro "*k*" é definido como corte. Para "*k*" igual "*-5*", somente os elementos com valores *DOI (x,y)* maiores ou iguais a "*-5*" tornaram-se visíveis. A Figura 4-18 ilustra a árvore filtrada.

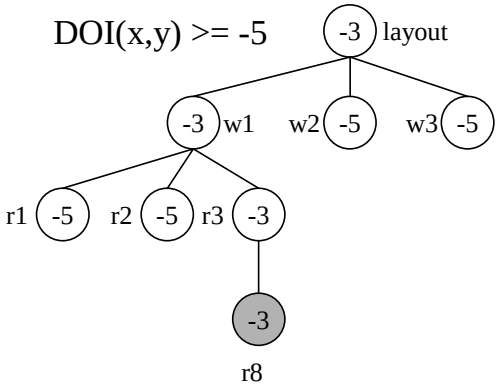

Figura 4-18 Elementos visualizados com $k \geq -5$