Transformada Imagem-Floresta (Estrutura de dados)

Prof. Dr. Paulo A. V. de Miranda

pmiranda@vision.ime.usp.br

Instituto de Matemática e Estatística (IME),

Universidade de São Paulo (USP)

Algoritmo da IFT

Algoritmo 1 — ALGORITMO GERAL DA IFT

ENTRADA: Imagem $\hat{I} = (\mathcal{D}_I, \vec{I})$, adjacência $\mathcal{A}$, e função de conexidade f.

SAÍDA: Imagens $\hat{P} = (\mathcal{D}_I, P)$ de predecessores, e $\hat{V} = (\mathcal{D}_I, V)$ de conexidade.

AUXILIARES: Fila de prioridade Q, variável tmp, e vetor de $estado$ inicialmente zerado.

1. ***Para Cada*** $t \in \mathcal{D}_I$, ***Faça*** $P(t) \leftarrow nil$ *e* $V(t) \leftarrow f(\langle t \rangle)$. ***Se*** $V(t) \neq +\infty$, ***Então*** *insira* t *em* Q.
2. ***Enquanto*** $Q \neq \emptyset$, ***Faça***
3. *Remova um pixel* s *de* Q *cujo valor* $V(s)$ *seja minimo.*
4. $estado(s) \leftarrow 1$.
5. ***Para Cada*** $t \in \mathcal{A}(s)$, *tal que* $estado(t) = 0$, ***Faça***
6. $tmp \leftarrow f(\pi_s \cdot \langle s, t \rangle)$.
7. ***Se*** $tmp < V(t)$, ***Então***
8. ***Se*** $V(t) \neq +\infty$, ***Então*** *remova* t *de* Q.
9. $P(t) \leftarrow s$, $V(t) \leftarrow tmp$, *e insira* t *em* Q.

Estrutura da fila de prioridade

A fila tem que suportar as três operações seguintes, de acordo com as linhas 3, 8 e 9 do algoritmo da IFT:

- remover um spel s de precedência máxima, ou seja com $V(s)$ mínimo (linha 3).
- remover um dado spel t de Q (linha 8).
- inserir um dado spel t em Q (linha 9).

Na verdade, essas operações dependem da estrutura exata da fila utilizada.

Estrutura da fila de prioridade

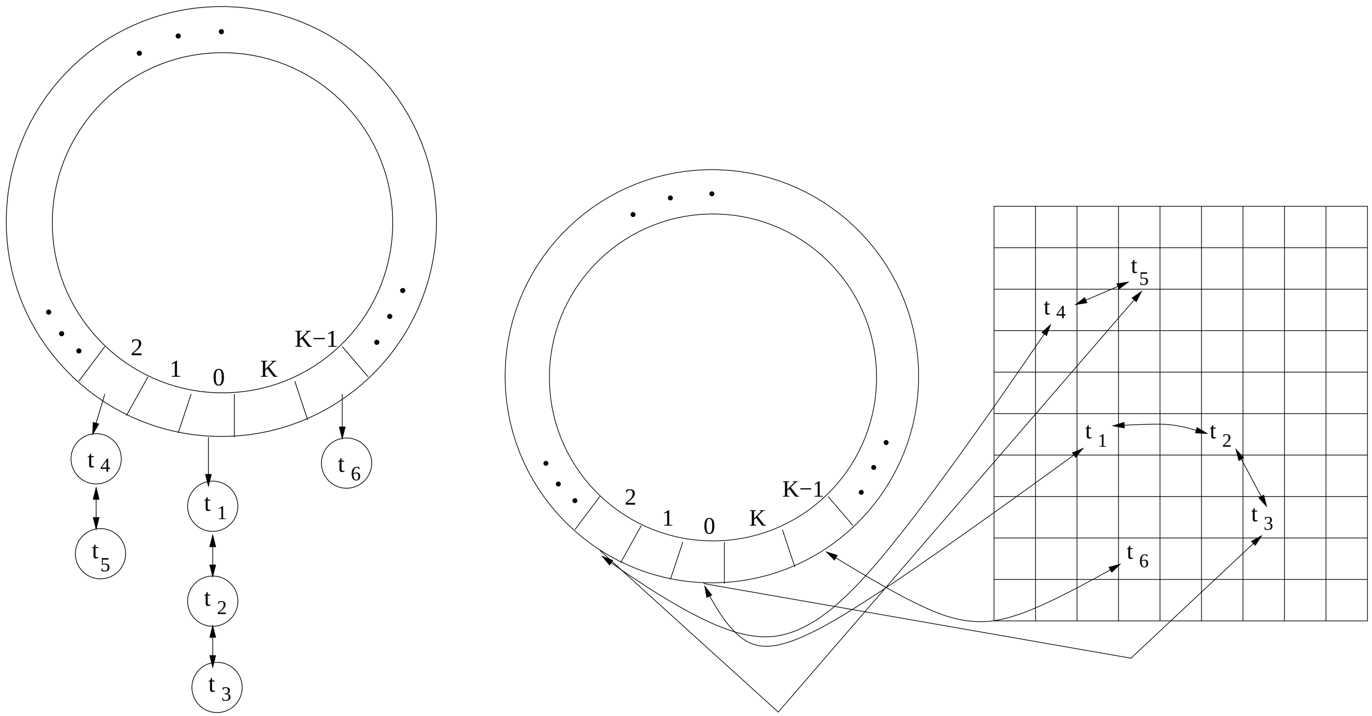
A implementação mais fácil para a fila Q usa um *heap binário*. Neste caso o algoritmo da IFT terá complexidade $O(M + N \log N)$, onde N é o número de nós (pixels) e M o número de arestas do grafo.

Estrutura da fila de prioridade

Na maioria das aplicações, porém, podemos usar funções de custo de caminho com incrementos de custo inteiros e limitados a uma constante K ao longo do caminho. Isto permite a utilização da fila circular de Dial com $K + 1$ posições.

- Cada posição i, $i = 0, 1, ..., K$, deve armazenar uma lista duplamente ligada de todos os pixels p com custo $V(p)$ tal que $i = V(p)\%(K + 1)$.

Estrutura da fila de prioridade



(a) Estrutura de Dial para a fila Q. (b) Estrutura proposta por A.X. Falcão. Como sabemos o tamanho máximo N do grafo, essas listas podem ser implementadas em uma única matriz X de ponteiros $X.next(p)$ e $X.prev(p)$ com N elementos.

Estrutura da fila de prioridade

A estrutura da fila proposta por Falcão possui:

- Um conjunto de $K + 1$ buckets (K = máximo incremento de custo), e
- Um vetor de N estruturas (N = tamanho da imagem), representando os spels inseridos na fila.

Assim, a estrutura de fila usa $O(K + N)$ de memória.

Estrutura da fila de prioridade

- Cada estrutura do segundo vetor corresponde a um nó da lista duplamente ligada (**DLL node**).
- Cada nó DLL tem dois ponteiros “Anterior” (**prev**) e “próximo” (**next**).
- Cada bucket do primeiro vetor aponta para o nó DLL do último spel inserido naquele bucket (os outros spels inseridos no mesmo bucket são recuperados, seguindo a lista ligada).

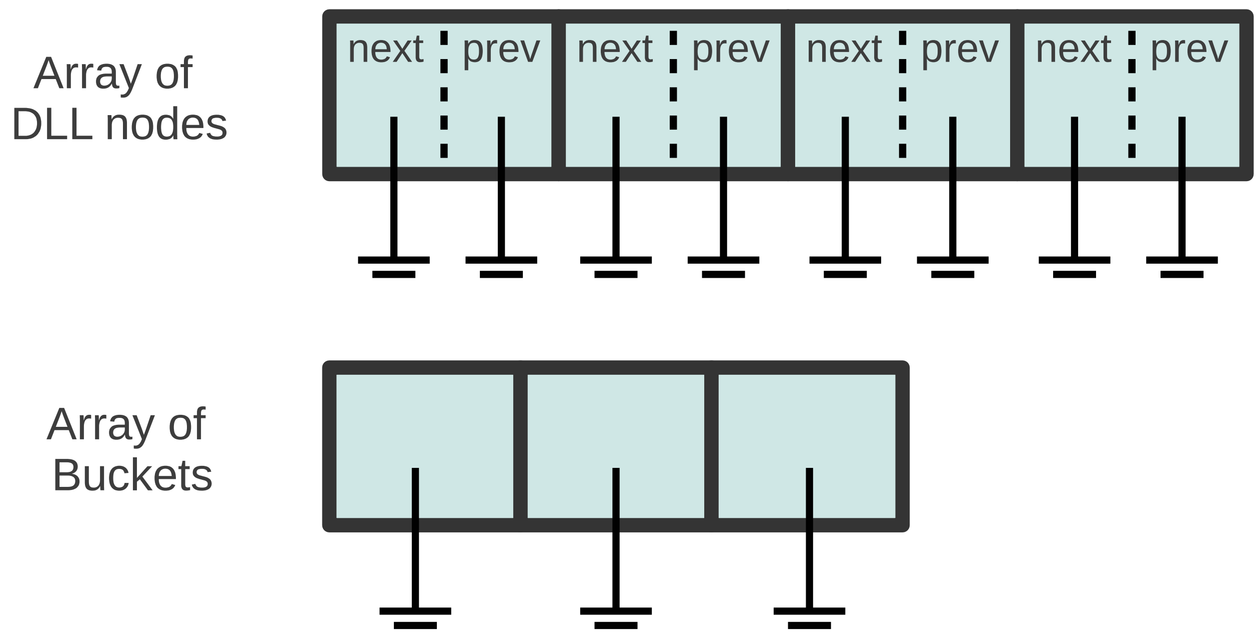
Estrutura da fila de prioridade

O nó DLL de um dado spel t pode ser obtido em tempo $O(1)$:

- O spel t tem um identificador (isto é, um número no intervalo $[0, N-1]$), que é utilizado como índice para acessar o seu nó DLL no primeiro array.

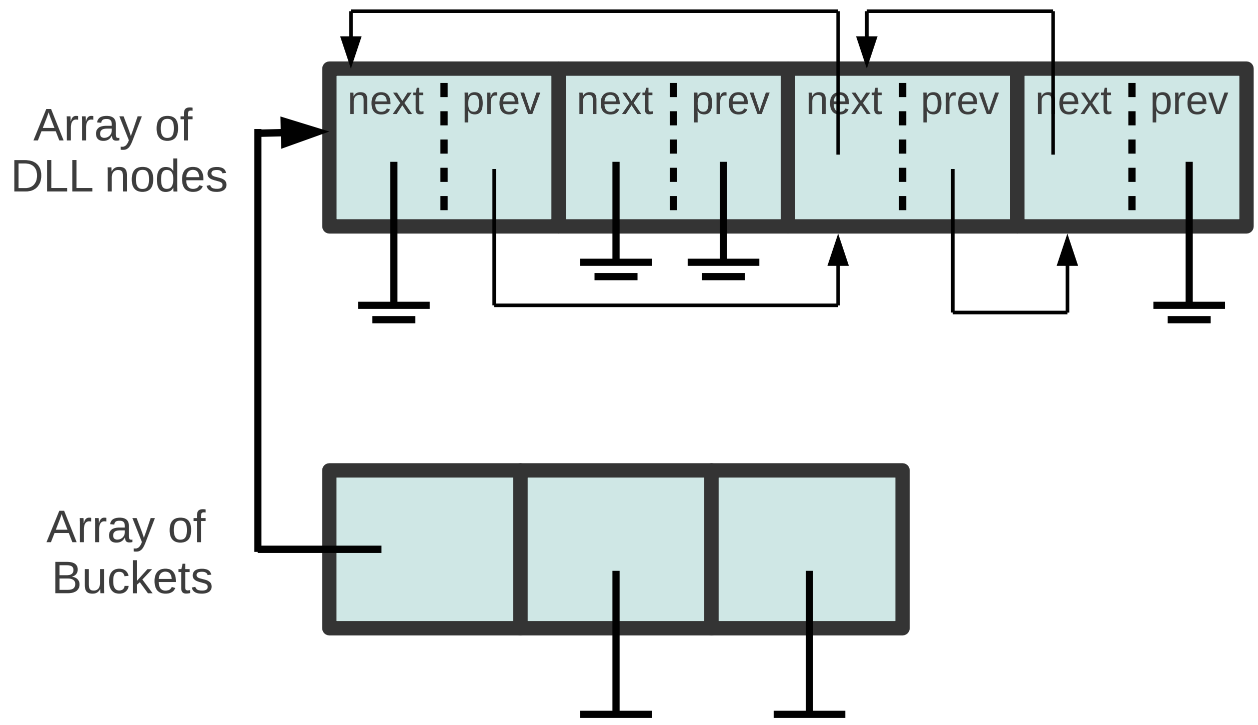
Estrutura da fila de prioridade

Vamos considerar o exemplo a seguir, onde $N = 4$ e $K = 2$. Uma fila vazia seria representada da seguinte forma:



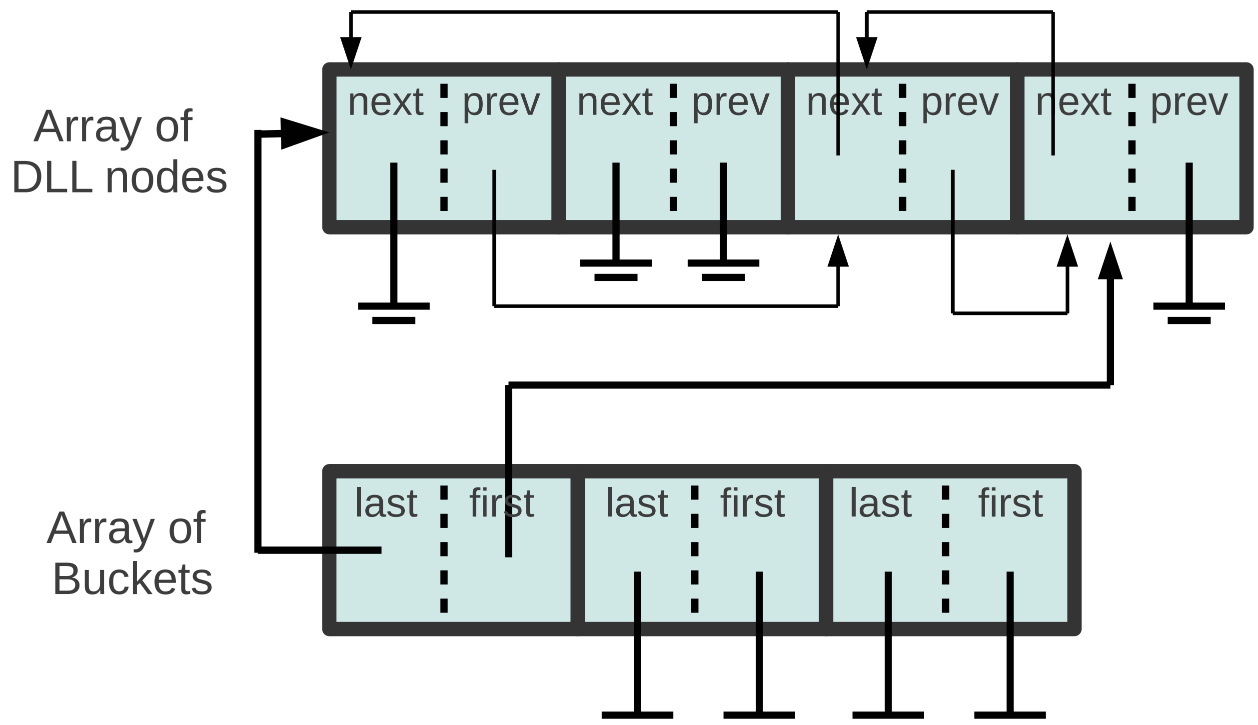
Estrutura da fila de prioridade

Os spels podem ser referenciados por seus índices no intervalo [0,3]. Agora, suponha que os spels 3,2 e 0 foram inseridos, nessa ordem, no primeiro bucket. Teremos:



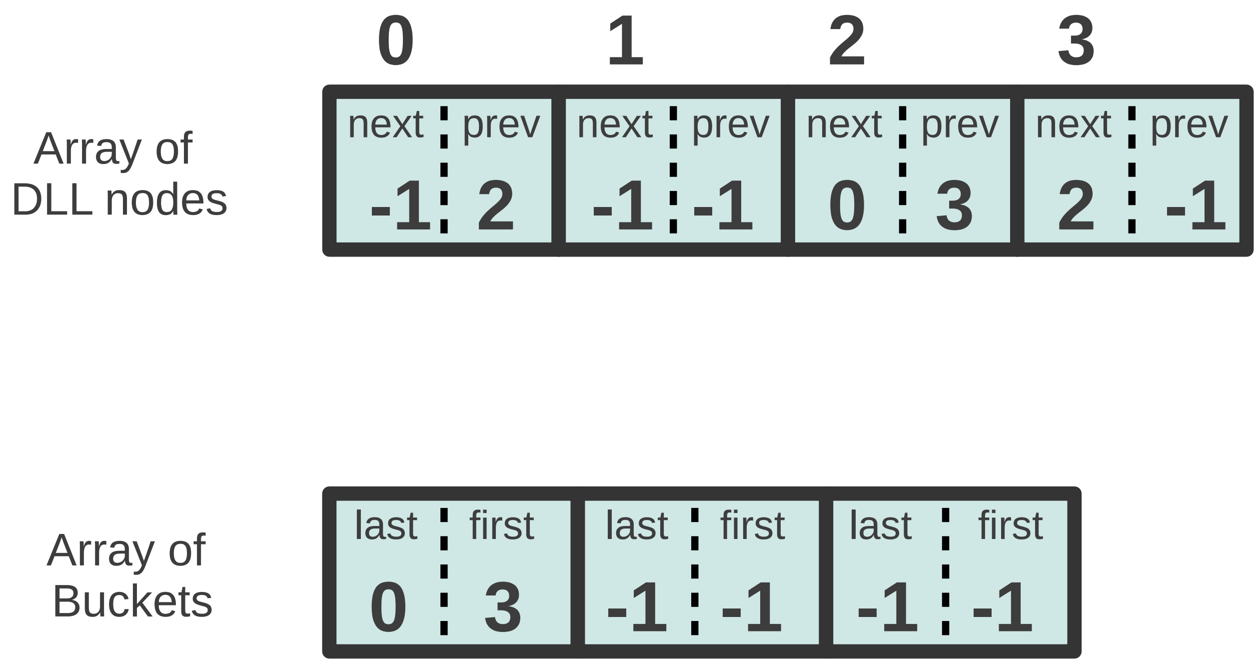
Estrutura da fila de prioridade

Na verdade, a fim de suportar inserção e remoção FIFO ou LIFO, nós geralmente consideramos também um apontador para o primeiro elemento em cada bucket:



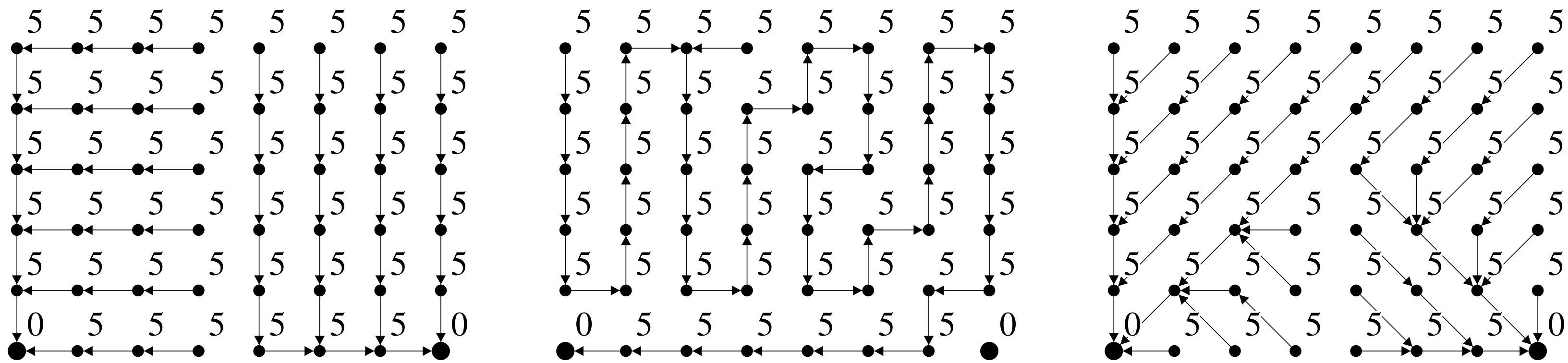
Estrutura da fila de prioridade

Poderíamos, também, representar a estrutura utilizando os índices dos spels em vez de usar ponteiros:



Resolvendo empates

O que fazer quando um pixel é alcançado por dois ou mais caminhos de mesmo custo?



Exemplos de *tie-breaking*. (a) Política FIFO. (b) Política LIFO. (c) Política FIFO com adjacência vizinhos-8.

Estrutura da fila de prioridade

- No caso da estrutura da fila proposta por Falcão, o algoritmo da IFT terá complexidade $O(M + N \times K)$.
- Para alguns casos particulares, a complexidade é ainda melhor $O(M + N + K)$ (e.g., função $f_{\max}$).
- Se a adjacência definir um grafo esparso, a IFT levará tempo proporcional ao número N de pixels, pois o número de pixels adjacentes pode ser considerado uma constante pequena.

Estrutura da fila de prioridade

- Note que a cada instante existe um valor mínimo V_{mim} e um valor máximo V_{max} de custo para os pixels armazenados em Q.
- A diferença $V_{max} - V_{min} \leq K$ deve ser mantida para garantir a corretude da fila.
- Em algumas aplicações sabemos que os incrementos são inteiros e limitados, mas não conhecemos o valor de K.
- Neste caso, a fila circular inicia com um dado tamanho K, mas antes de inserir um novo pixel devemos verificar a necessidade de realocar ou não mais elementos para a fila.