Listas de Prioridades Cinéticas

Tese de Mestrado de:

Guilherme Dias da Fonseca

(bolsista CAPES)

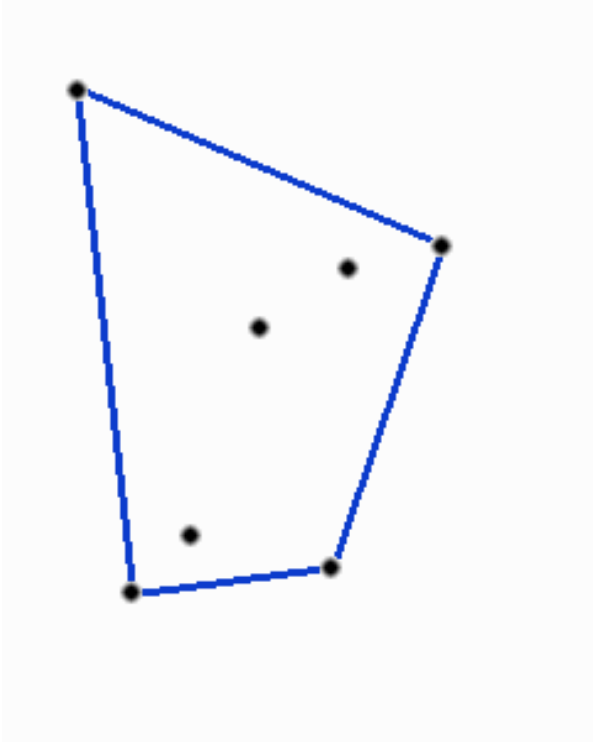
Orientadora:

Celina M. H. de Figueiredo

03/2003

Introdução

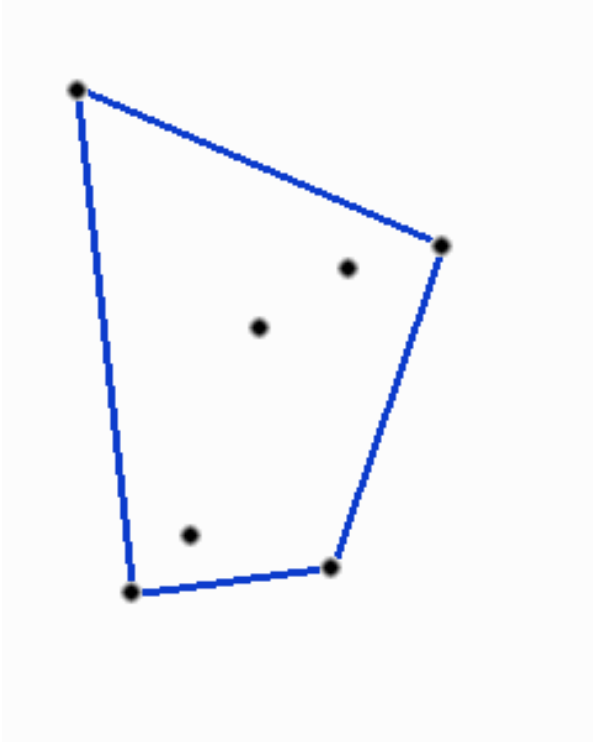
Estruturas de Dados Cinéticas



- **Fecho convexo:**
  Menor polígono convexo envolvendo um conjunto de pontos.

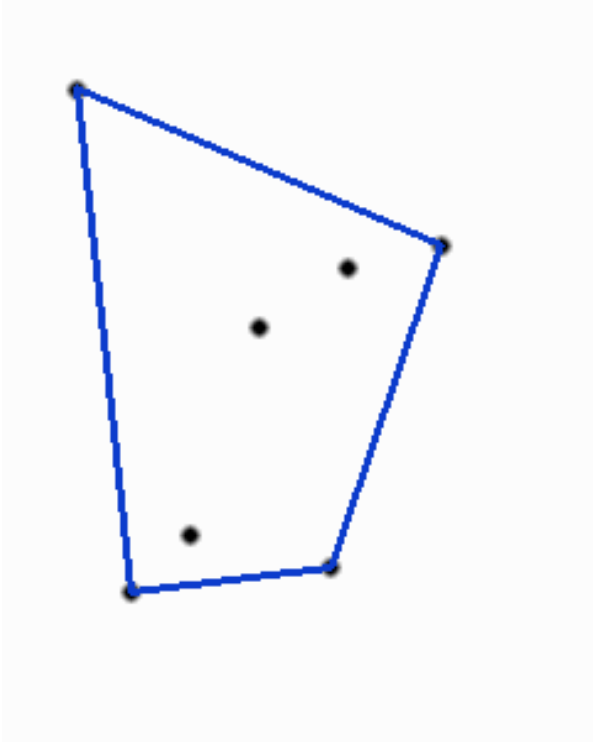
E se os pontos estiverem em movimento???

Estruturas de Dados Cinéticas



- Outras alternativas
  - Repetições
  - Dimensão adicional
- Operações das EDC:
  - Inicializar
  - Consultar atributo
  - Avançar no tempo
  - Alterar trajetória

Problemas Dinâmicos



- **Pontos podem ser inseridos e removidos**
- Novas operações:
  - Inserir
  - Remover

Um Problema Mais Simples

- Determinar o ponto extremo em uma direção
- Problema unidimensional
- Equivalente a:
  Determinar o maior valor em um conjunto de números reais

Envelope Superior

- **<u>Envelope superior:</u>** sequência de funções máximas
- Estamos computando um envelope superior, porém as **trajetórias não são previamente definidas**
- Paradigma de **linha de varredura**

Envelope Superior

- Máximo de segmentos no envelope superior de *n* funções onde cada par de funções se intercepta no máximo δ vezes:

$$\lambda_{\delta}(n)$$

- Se as funções são definidas apenas em um intervalo:

$$\lambda_{\delta+2}(n)$$

Envelope Superior

- Máximo de segmentos no envelope superior de n funções onde cada par de funções se intercepta no máximo δ vezes:

$$\lambda_{\delta}(n)$$

- Se as funções são definidas apenas em um intervalo:

$$\lambda_{\delta+2}(n)$$

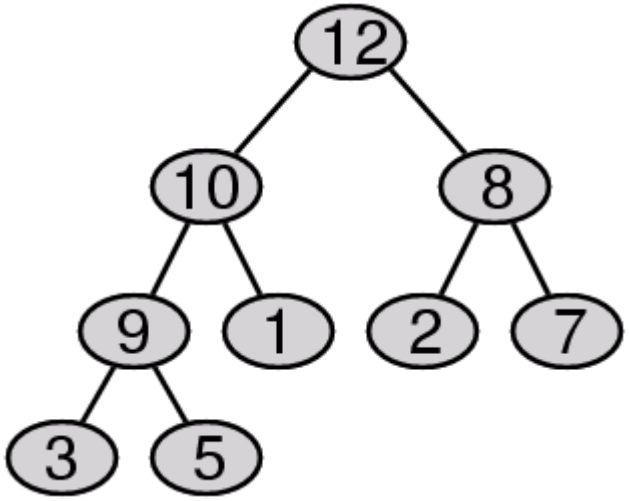
- $\lambda_1(n)=n$
- $\lambda_2(n)=2n-1$
- $\lambda_3(n)=\Theta(n\ \alpha(n))$
- $\lambda_{\delta}(n)$ é quase linear em n, com δ constante

Listas de Prioridades Cinéticas

- Determinam o elemento máximo
- Operações:
  - Inicializar
  - Inserir
  - Remover
  - Avançar no tempo
  - Determinar máximo
  - Alterar trajetória
- Estruturas:
  - Heap Cinético
  - Torneio Cinético
  - Heater Cinético
  - Hanger Cinético
  - Redução a Fecho Convexo Dinâmico

Heap Cinético

Heap

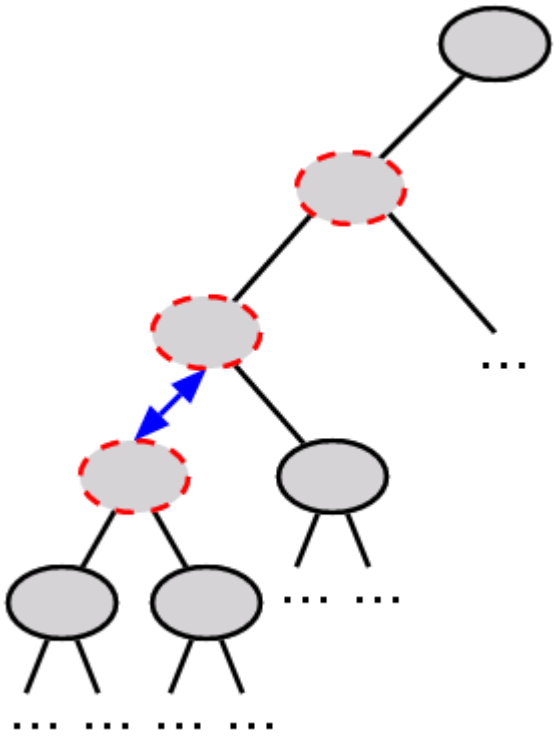


- Árvore binária cheia onde todo vértice é maior que seus descendentes
- Suporta inserção e remoção em tempo O(lg *n*)

E se os valores mudarem continuamente?

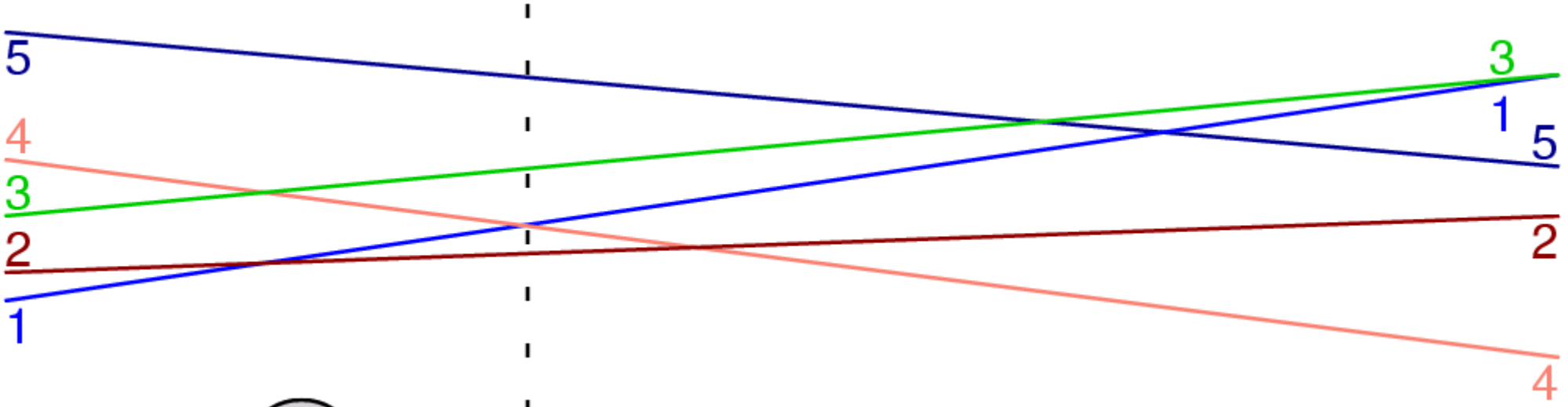
- Necessidade de trocar a posição de elementos

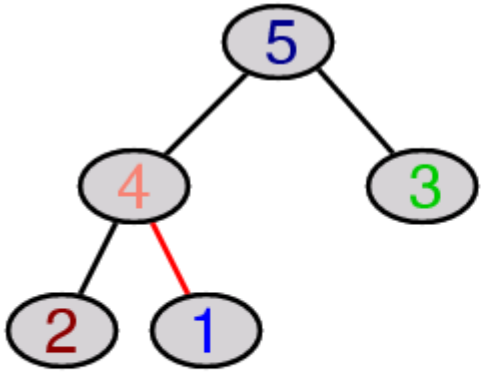
Fila de Eventos



- Agendamos um evento para cada nó interno
- O evento é agendado no primeiro instante em que o nó intercepta um de seus filhos
- Processar evento (no heap):
  - Trocar os dois vértices
  - Reagendar até 3 eventos

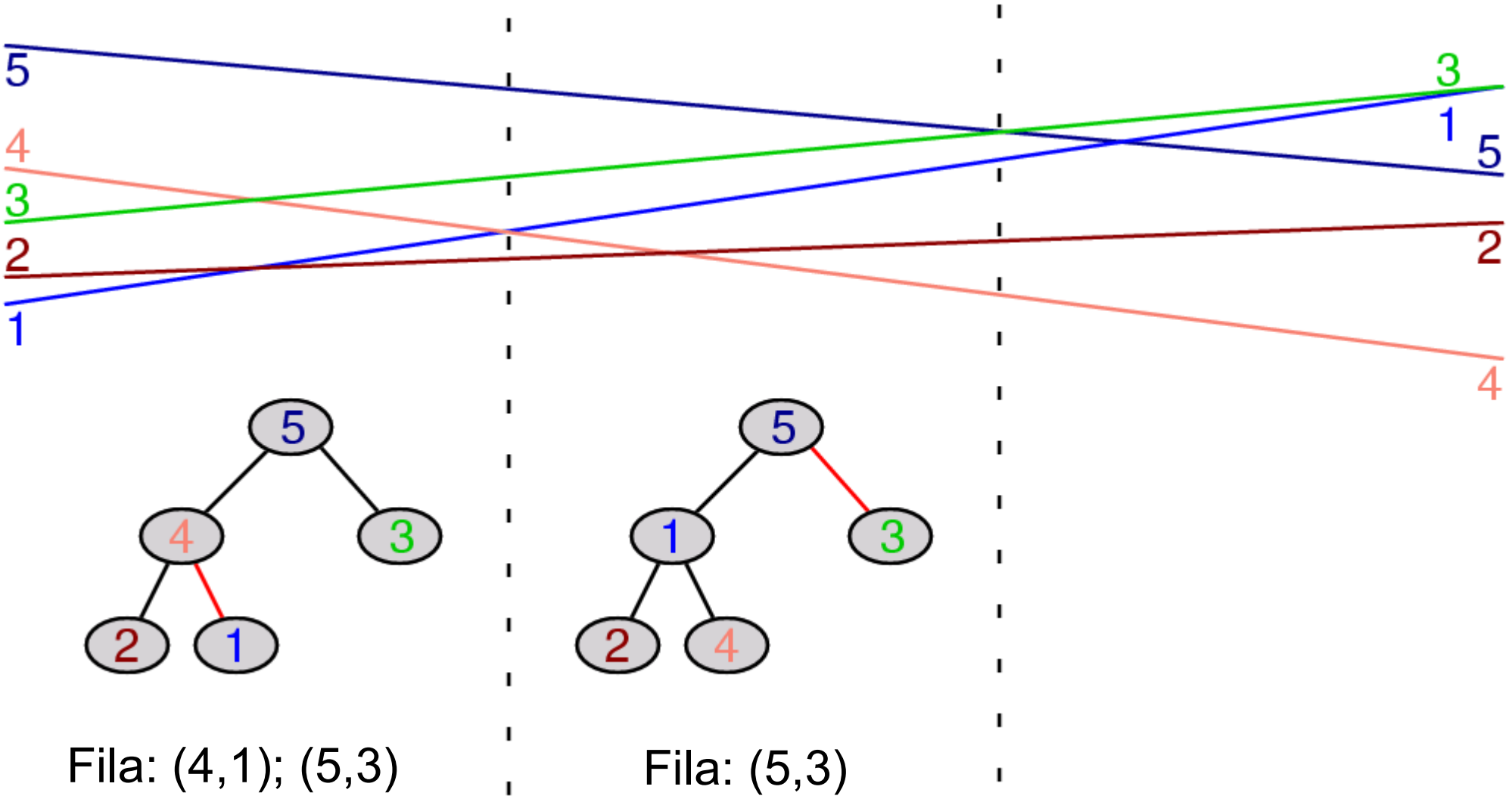
Heap Cinético



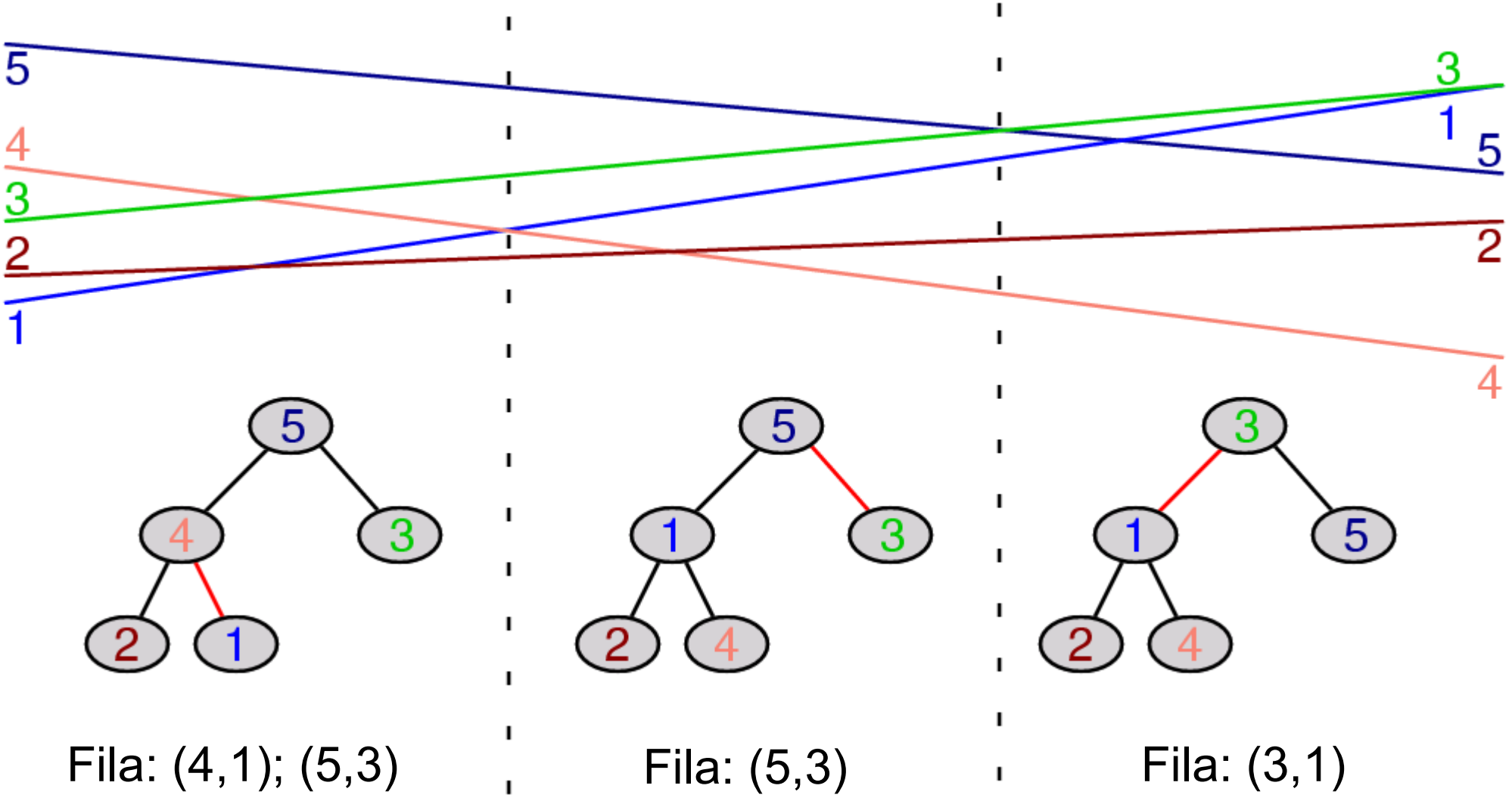


Fila: (4,1); (5,3)

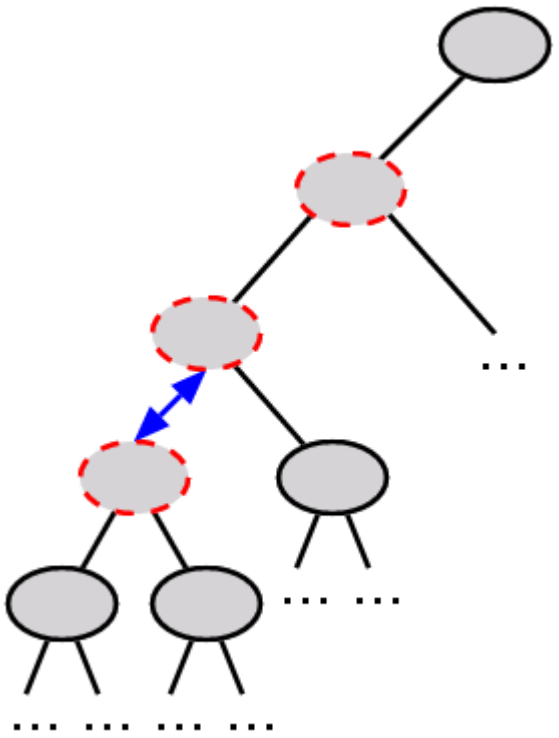
Heap Cinético



Heap Cinético

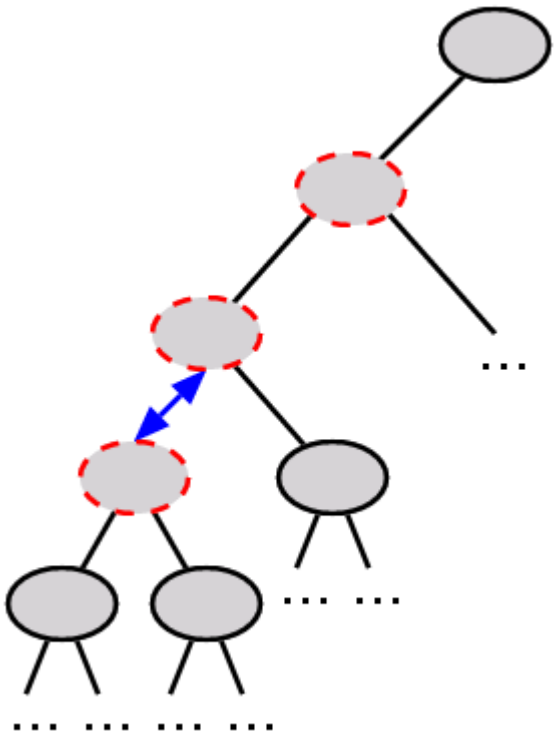


Análise de Complexidade



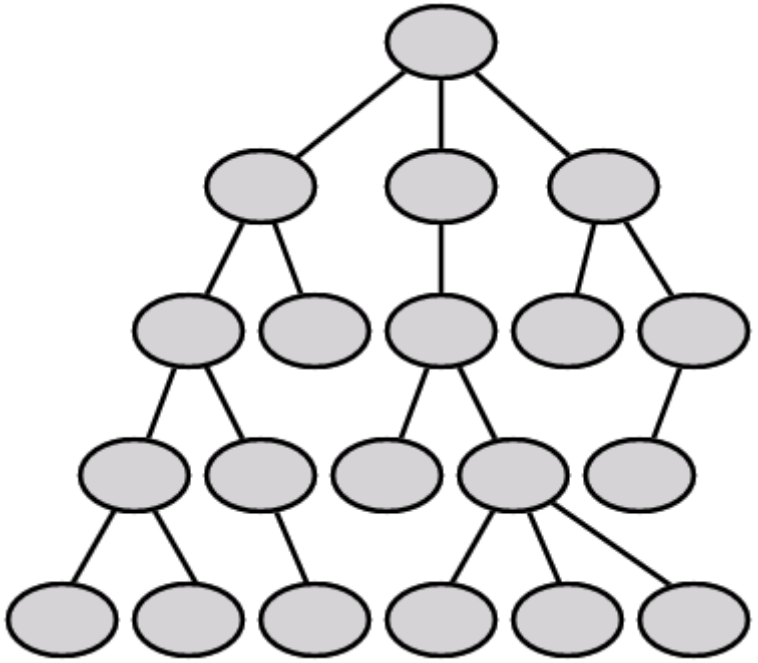
- Quanto tempo é gasto processando cada evento?
- Resposta: O(lg *n*), implementando a fila como um heap ou similar
- Quantos eventos são processados?
- Resposta: Desconhecida.

Análise de Complexidade



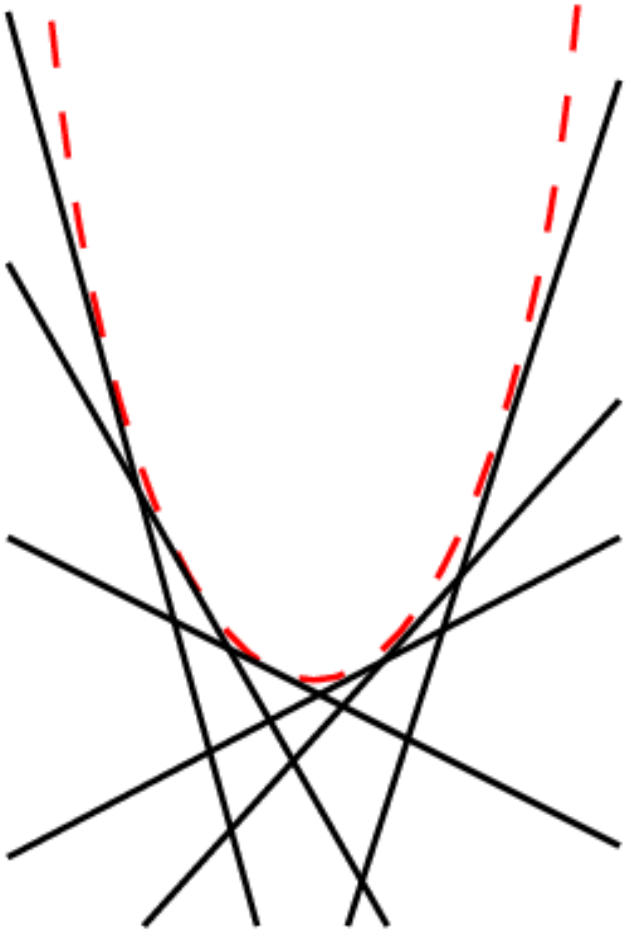
- No máximo quantos eventos são processados caso
  1. As funções sejam lineares
  2. Nenhum elemento seja inserido ou removido
- Resposta: A soma dos níveis dos elementos

Análise de Complexidade



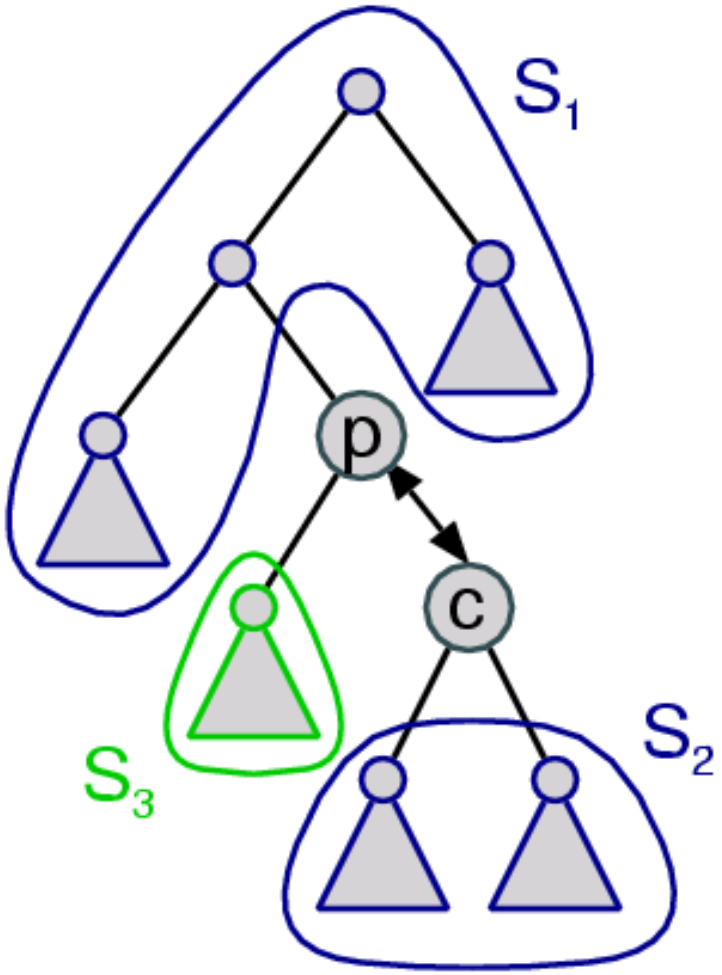
Nível		Nós		
0	x	1	=	0
1	x	3	=	3
2	x	5	=	10
3	x	5	=	15
4	x	6	=	24
				52 eventos

Limite Inferior



- Retas tangentes a parábola
- Cada elemento aparece uma vez no envelope superior (ponto de tangência)
- Cada evento sobe exatamente um elemento em exatamente um nível

Limite Superior



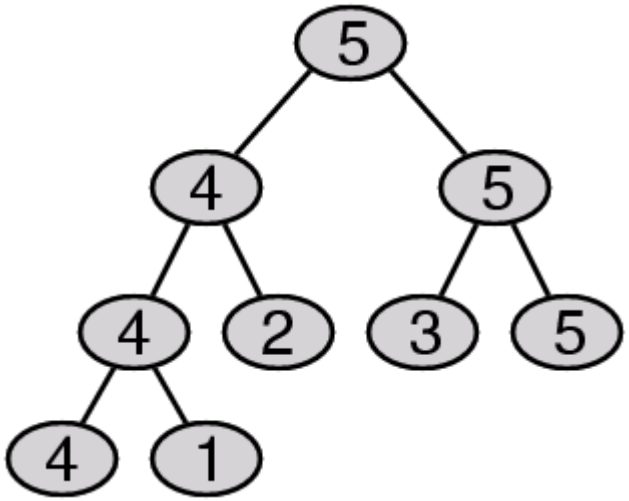
- Função potencial:
  $\Phi(e, t)$ = número de ancestrais de e, no tempo t, que interceptam e após t
- Obs: $\Phi(e, t) \leq$ nível de e
- $\Phi(e_{1,2}, t+) = \Phi(e_{1,2}, t-)$
- $\Phi(e_3, t+) \leq \Phi(e_3, t-)$
- $\Phi(p, t+) = \Phi(p, t-)$
- $\Phi(c, t+) = \Phi(c, t-) - 1$

Complexidade de Tempo

- Sem inserções e remoções:
  - Retas (ou pseudo-retas):
    Eventos: $\Theta(n \lg n)$
    Tempo: $O(n \lg^2 n)$
  - Versão $(\lg n)$-ária:
    Eventos: $\Theta(n \lg n / \lg \lg n)$
    Tempo: $O(n \lg^2 n / \lg \lg n)$
- Com inserções e remoções:
  - Retas (ou pseudo-retas):
    Eventos: $O(n^{3/2} (\lg n)^{1/2})$, $\Omega(n \lg n)$
- Localidade: O(1)

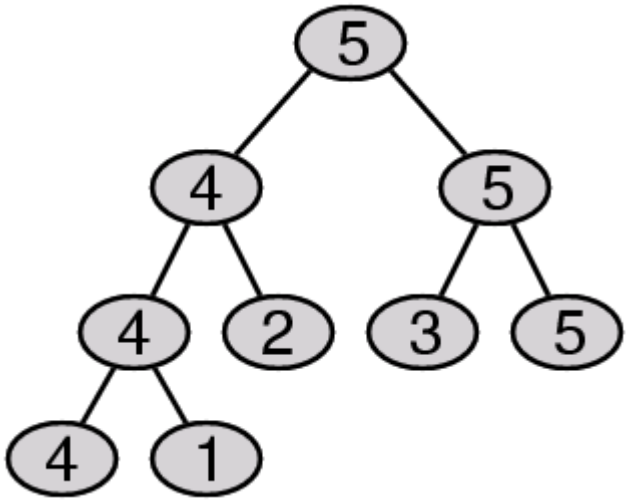
Torneio Cinético

Torneio



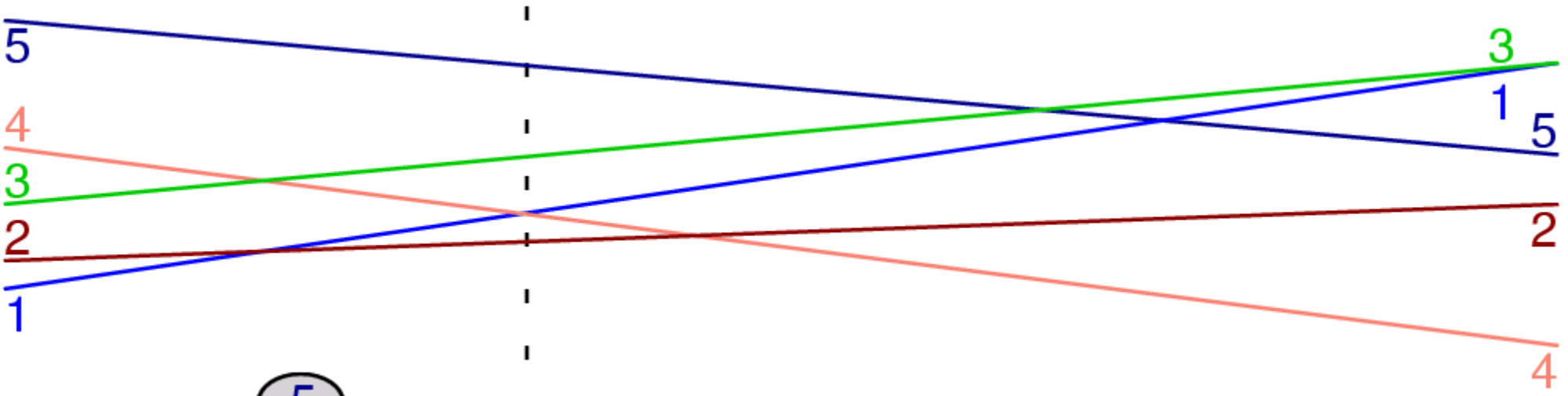
- Algoritmo estático de divisão e conquista
- Elementos nas folhas
- Cada nó interno contém seu maior filho

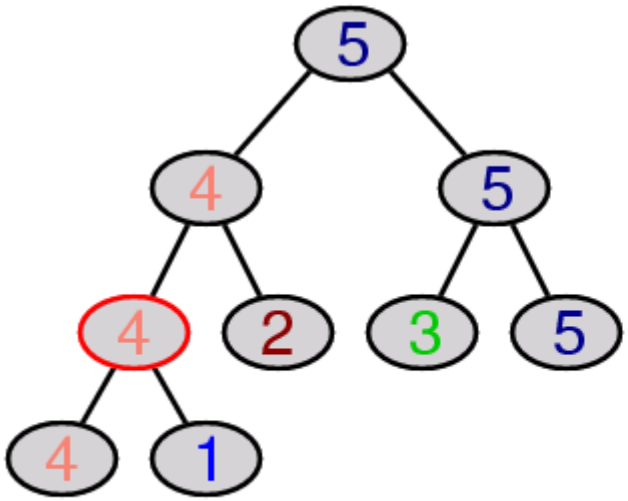
Torneio Cinético



- Agenda-se um evento para cada nó interno no próximo momento em que um filho intercepta seu pai
- Ao processar um evento, até O(lg *n*) eventos podem ser reagendados
- Localidade: O(lg *n*)

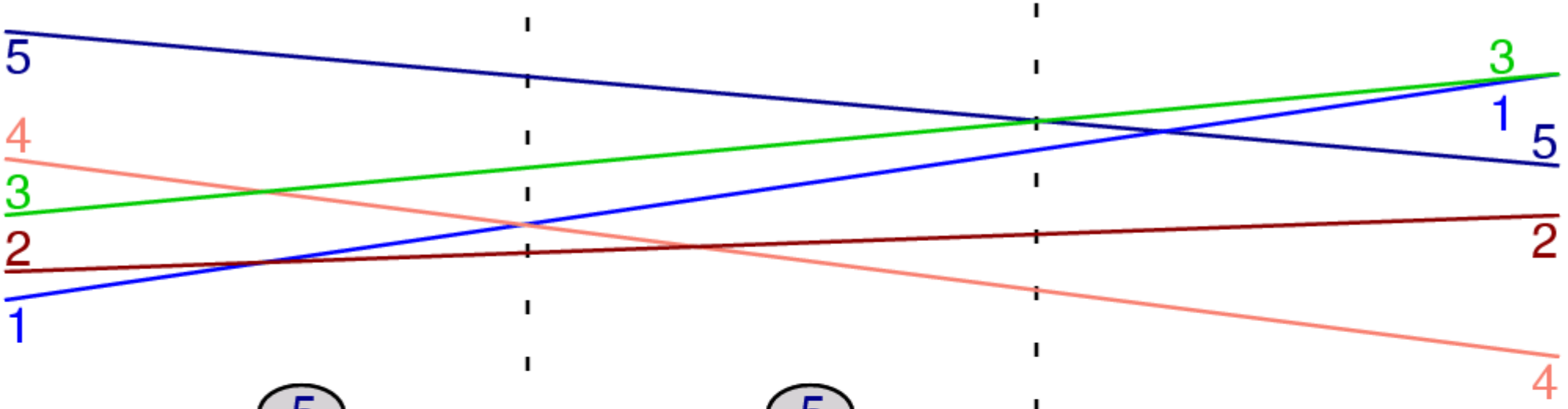
Torneio Cinético

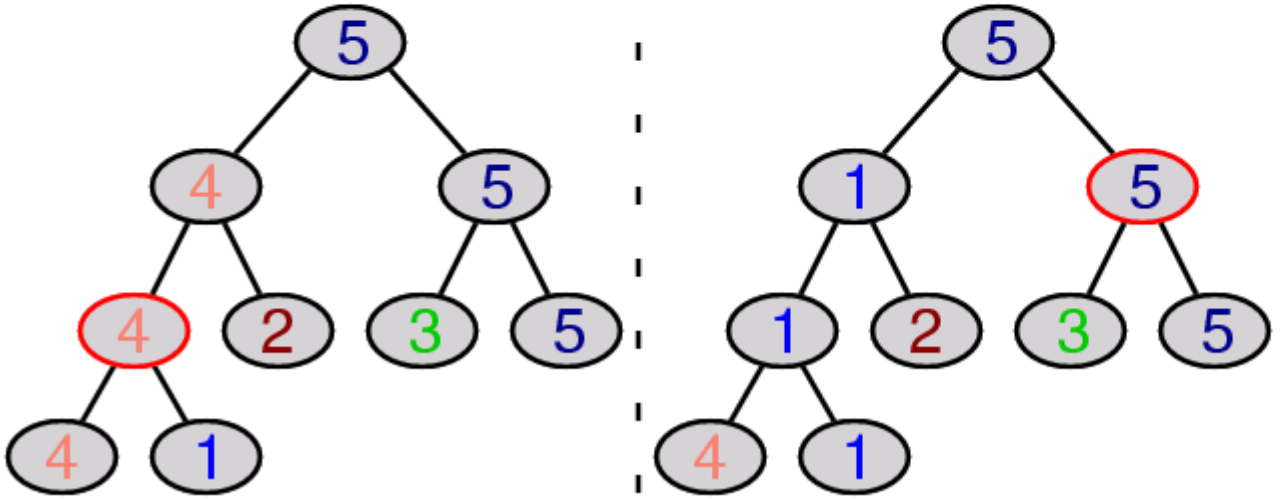




Fila: (4,1); (4,2); (3,5)

Torneio Cinético

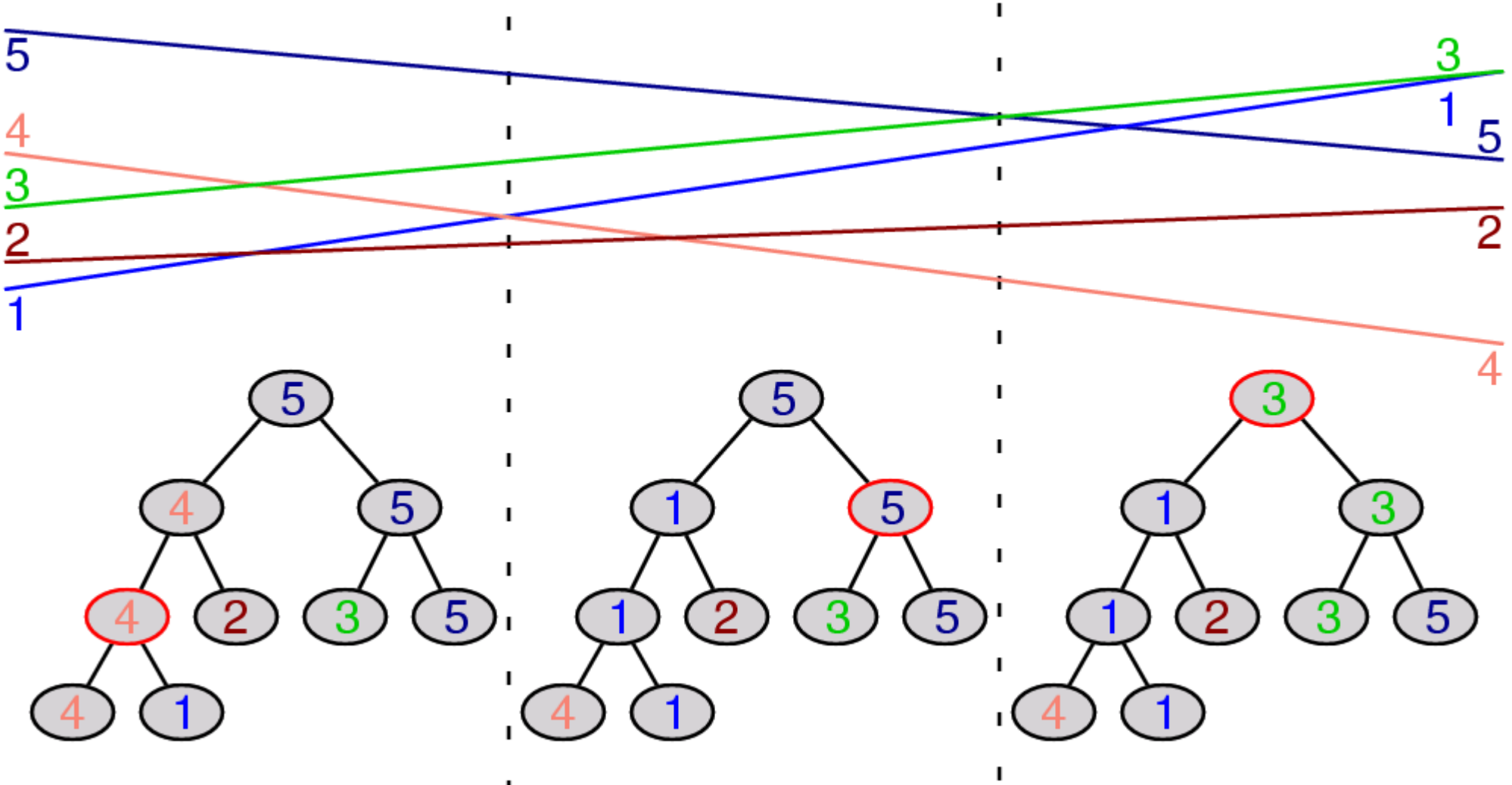




Fila: (4,1); (4,2); (3,5)

Fila: (3,5); (1,5)

Torneio Cinético

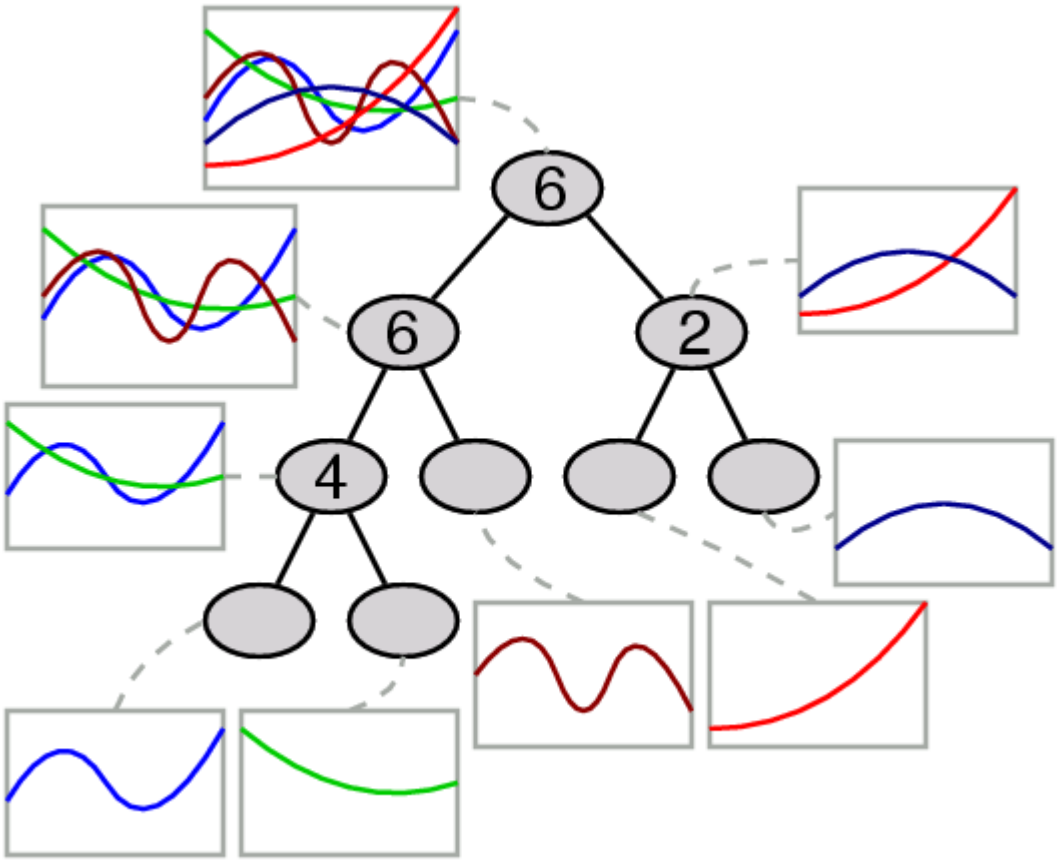


Fila: (4,1); (4,2); (3,5)

Fila: (3,5); (1,5)

Fila: (1,3)

Análise de Complexidade

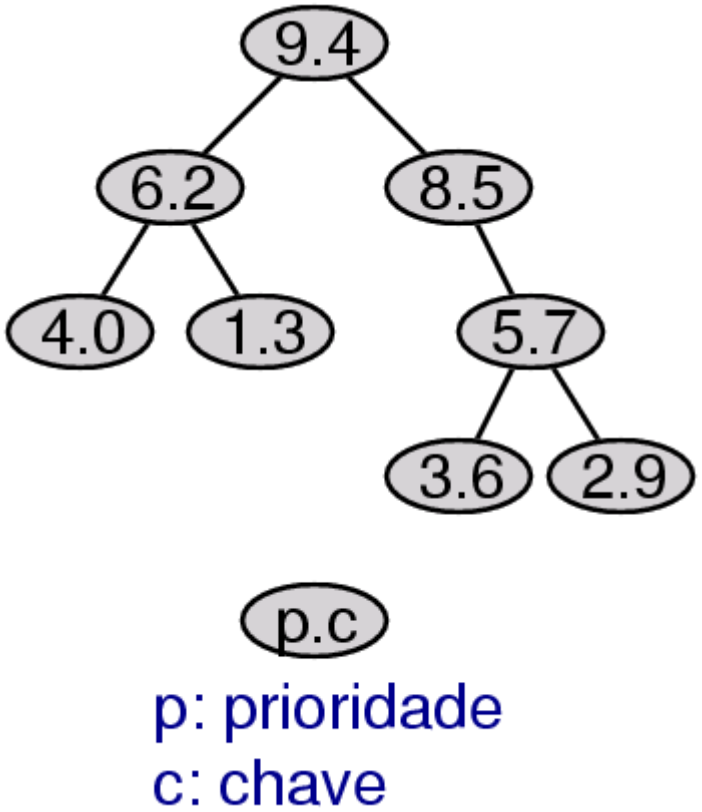


- Cada alteração leva tempo $O(\lg n)$

- Alterações na árvore: $O(\lambda_{\delta}(n) \lg n)$
- Com inserções e remoções: $O(\lambda_{\delta+2}(n) \lg n)$
- Com retas, sem inserções e remoções: $O(n \lg n)$
- Com retas, com inserções e remoções: $O(n \lg n\, \alpha(n))$

Heater Cinético

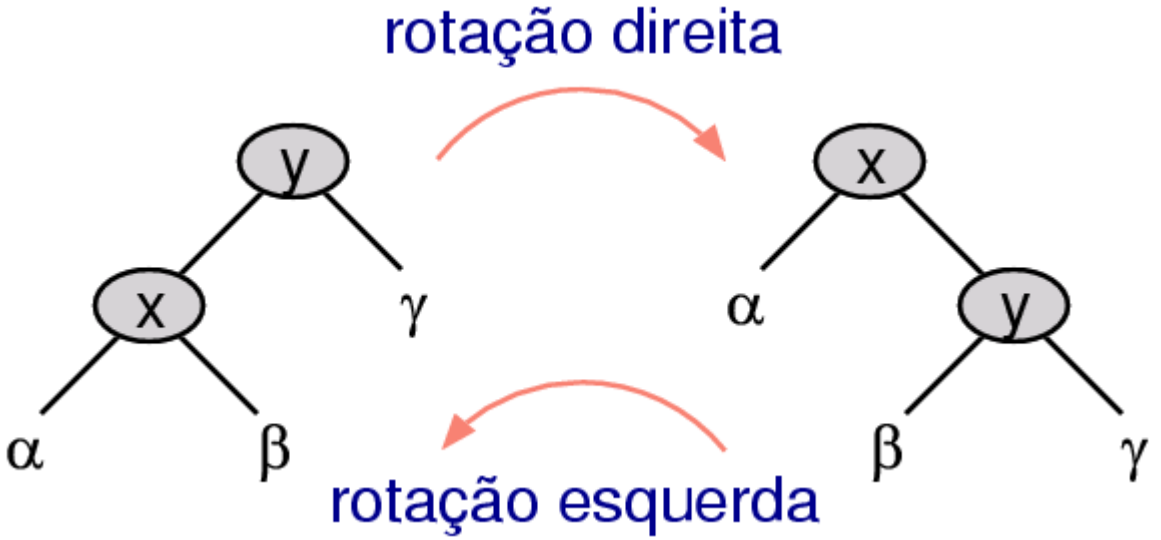
Treap



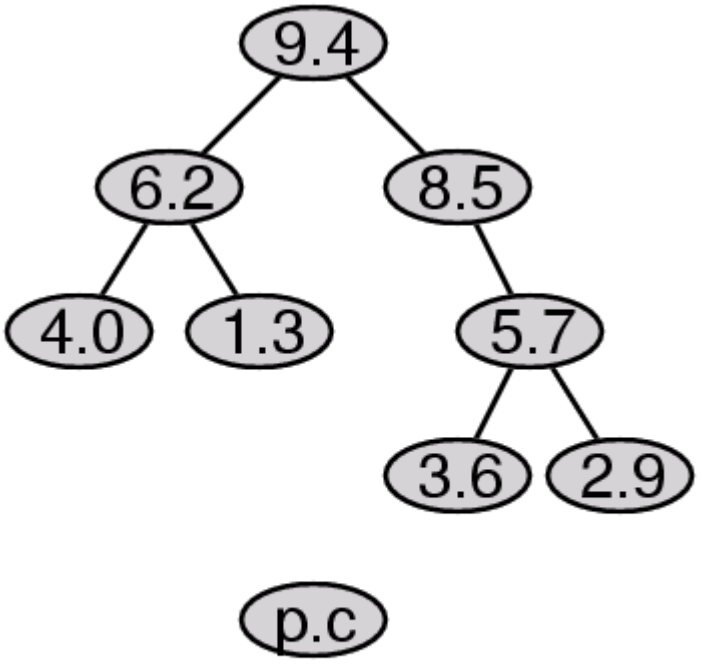
- Ordenação de heap com relação a prioridade
- Árvore binária de busca com relação a chave
- Existe e é único
- Não necessariamente balanceado
- Se as prioridades forem números reais aleatórios, a altura esperada é O(lg *n*)

Heater

- As chaves são aleatórias
- A fila de eventos é igual a do heap
- Para inserir ou remover elementos, assim como processar eventos, são necessárias rotações
- Localidade: O(1)



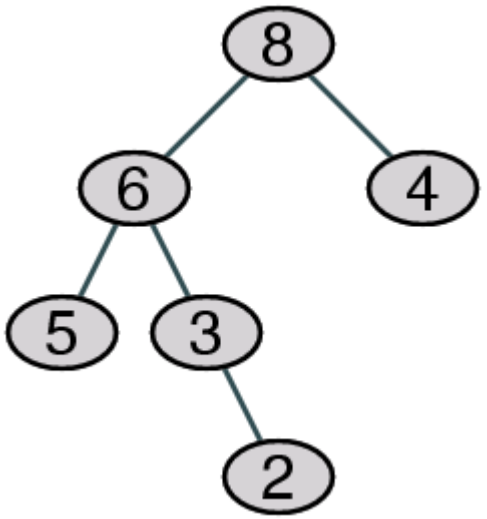
Análise de Complexidade



- Probabilidade do *i*-ésimo maior elemento ser pai do (*i*+1)-ésimo maior elemento é O(1/*i*)
- Pois isto só ocorre se as chaves de *i* e *i*+1 forem consecutivas dentre as chaves de 1 a *i*+1
- Complexidades de tempo iguais as do torneio cinético, porém com localidade O(1) e randomizadas

Hanger Cinético

Hanger



- Balanceamento randomizado
- Não necessita de rotações
- Fila de eventos e processamento de eventos iguais ao do heap
- Algoritmos simples
- Análise delicada
- Complexidades iguais ao heater e torneio

Hanger

- **Inserir 6**
- Inserir 3
- Inserir 2
- Inserir 4
- Inserir 5
- Inserir 8
- Remover 8

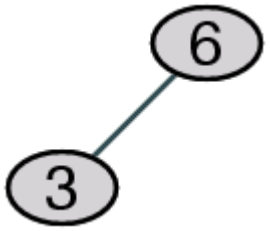
Bits Aleatórios: 001101100

Hanger

- Inserir 6
- **Inserir 3**
- Inserir 2
- Inserir 4
- Inserir 5
- Inserir 8
- Remover 8

Bits Aleatórios: 0011011100

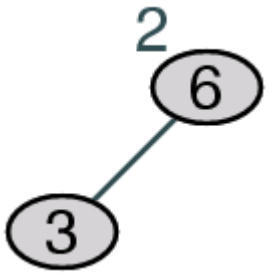
Hanger



- Inserir 6
- **Inserir 3**
- Inserir 2
- Inserir 4
- Inserir 5
- Inserir 8
- Remover 8

Bits Aleatórios: 001101100

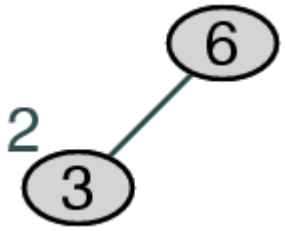
Hanger



- Inserir 6
- Inserir 3
- **Inserir 2**
- Inserir 4
- Inserir 5
- Inserir 8
- Remover 8

Bits Aleatórios: 001101100

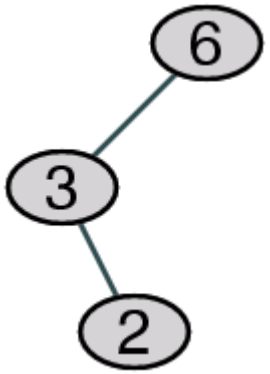
Hanger



- Inserir 6
- Inserir 3
- **Inserir 2**
- Inserir 4
- Inserir 5
- Inserir 8
- Remover 8

Bits Aleatórios: 0011101100

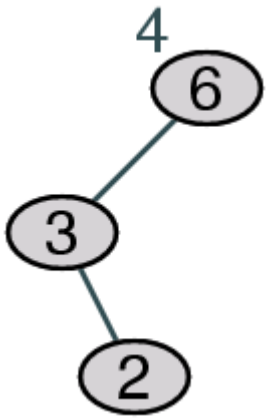
Hanger



- Inserir 6
- Inserir 3
- **Inserir 2**
- Inserir 4
- Inserir 5
- Inserir 8
- Remover 8

Bits Aleatórios: 001101100

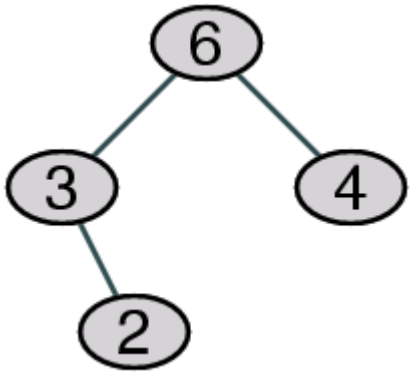
Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- **Inserir 4**
- Inserir 5
- Inserir 8
- Remover 8

Bits Aleatórios: 0011101100

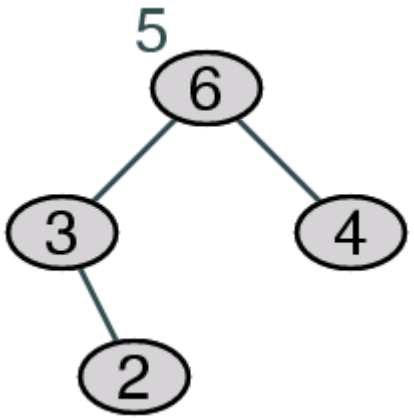
Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- **Inserir 4**
- Inserir 5
- Inserir 8
- Remover 8

Bits Aleatórios: 0011**01100**

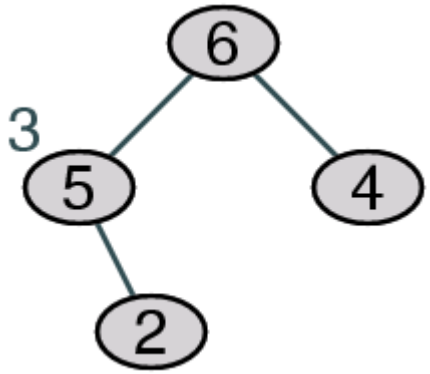
Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- Inserir 4
- **Inserir 5**
- Inserir 8
- Remover 8

Bits Aleatórios: 0011101100

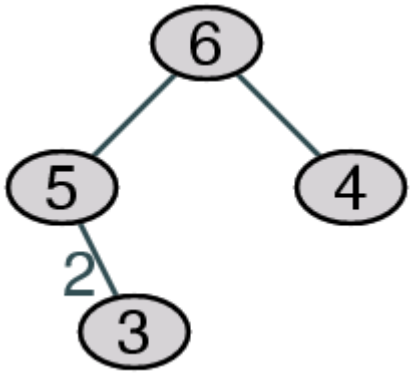
Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- Inserir 4
- **Inserir 5**
- Inserir 8
- Remover 8

Bits Aleatórios: 0011011100

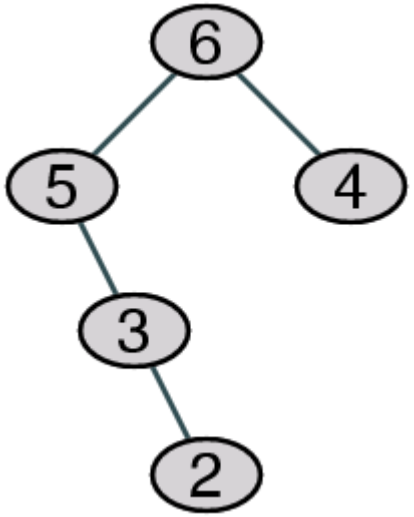
Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- Inserir 4
- **Inserir 5**
- Inserir 8
- Remover 8

Bits Aleatórios: 0011011100

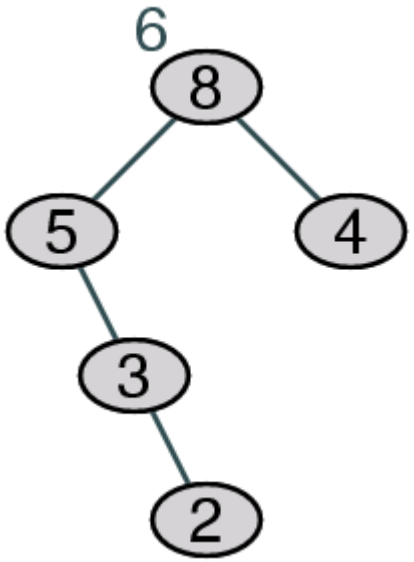
Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- Inserir 4
- **Inserir 5**
- Inserir 8
- Remover 8

Bits Aleatórios: 001101100

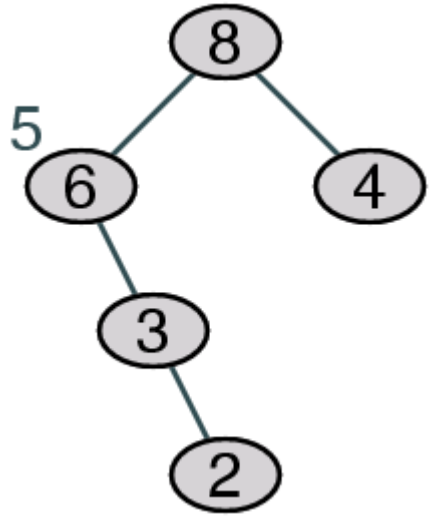
Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- Inserir 4
- Inserir 5
- **Inserir 8**
- Remover 8

Bits Aleatórios: 001101100

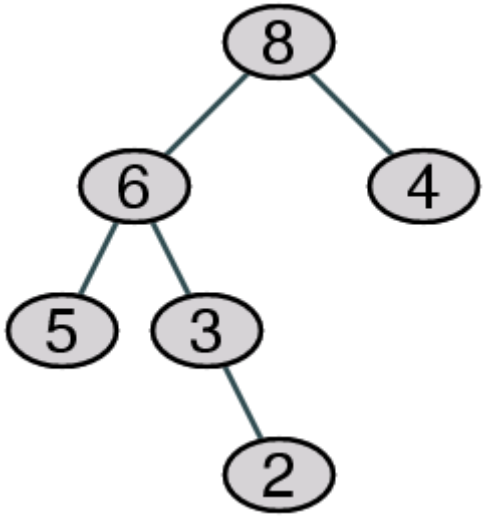
Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- Inserir 4
- Inserir 5
- **Inserir 8**
- Remover 8

Bits Aleatórios: 001101100

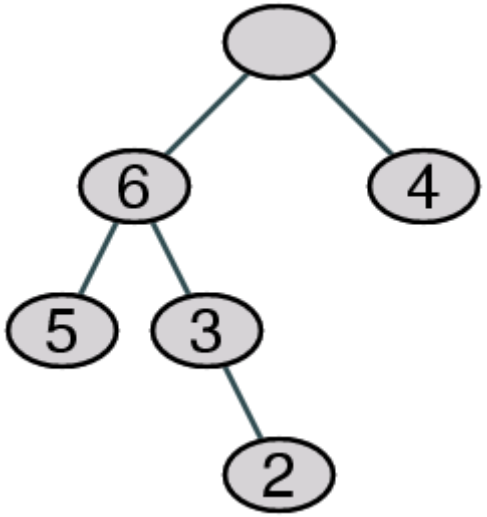
Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- Inserir 4
- Inserir 5
- **Inserir 8**
- Remover 8

Bits Aleatórios: 001101100

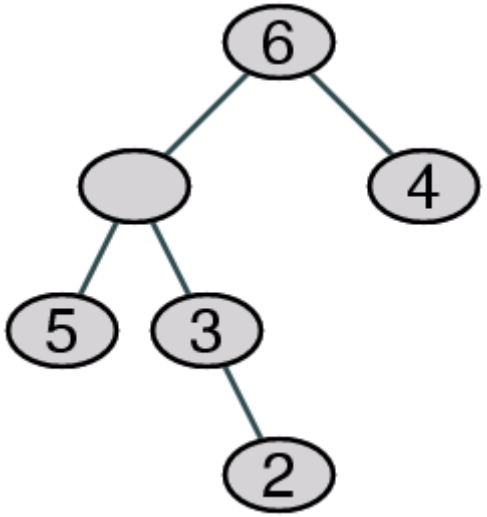
Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- Inserir 4
- Inserir 5
- Inserir 8
- **Remover 8**

Bits Aleatórios: 001101100

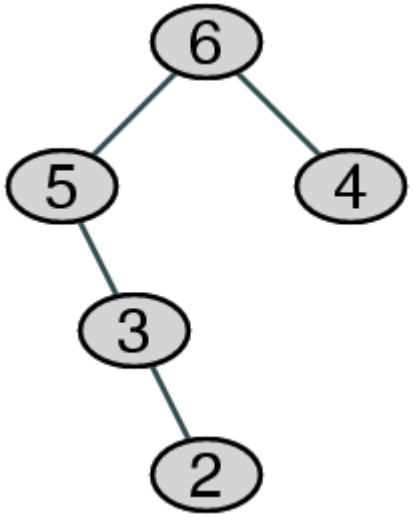
Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- Inserir 4
- Inserir 5
- Inserir 8
- **Remover 8**

Bits Aleatórios: 001101100

Hanger



- Inserir 6
- Inserir 3
- Inserir 2
- Inserir 4
- Inserir 5
- Inserir 8
- **Remover 8**

Bits Aleatórios: 001101100

Análise de Complexidade

- Nível esperado do *n*-ésimo elemento é O(lg *n*)
- Prova por indução em *n*, a partir de recorrência com coeficientes binomiais:

$$E\left(2^{L_n}\right) = \sum_{i=0}^{n-2} \binom{n-2}{i} \frac{1}{2^{n-2}} 2E\left(2^{L_{i+1}}\right)$$

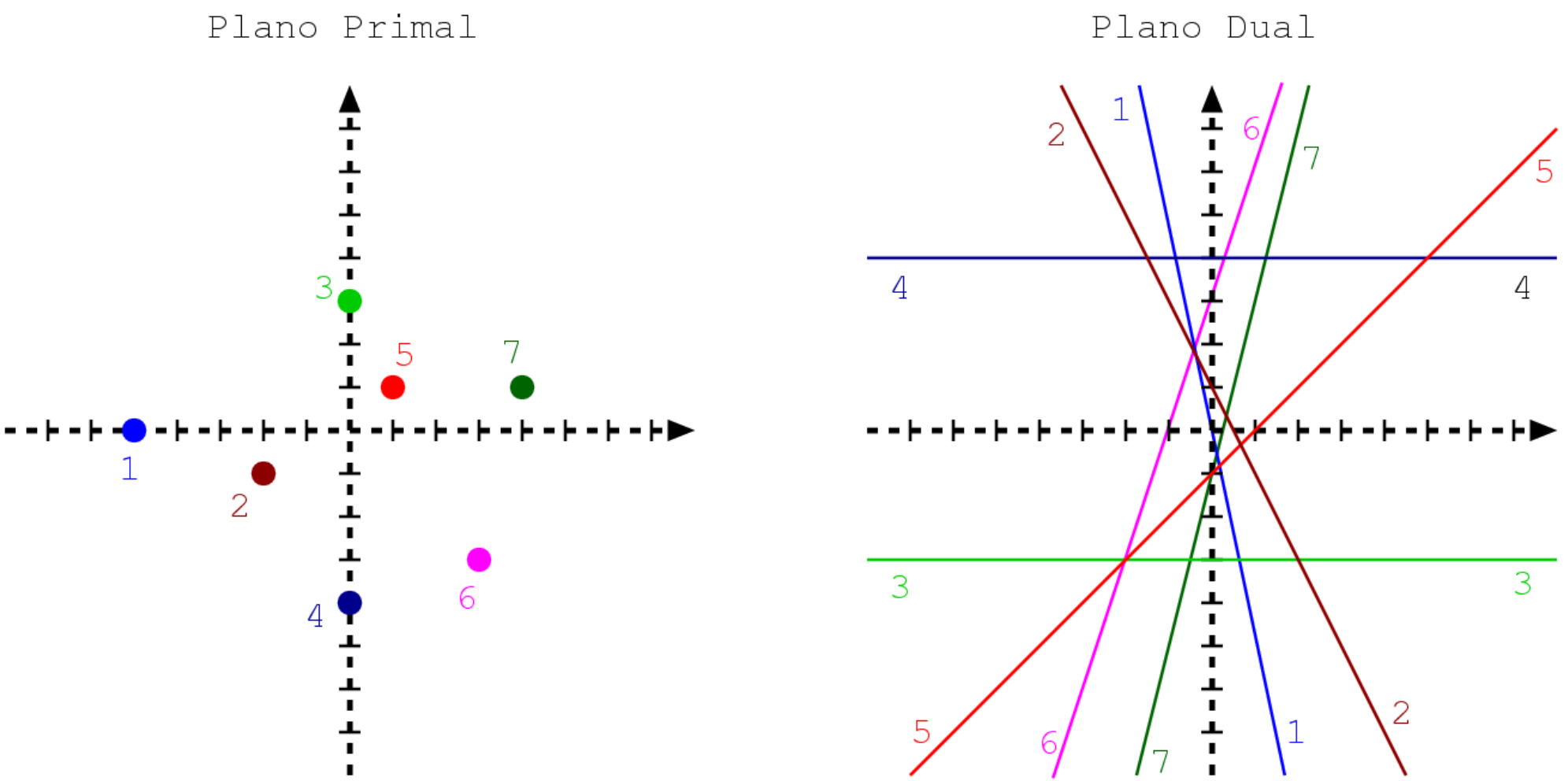
- Partindo de E(L_n)=O(lg *n*), provamos que a probabilidade do *n*-ésimo elemento ser pai do (*n*+1)-ésimo elemento é O(1/*n*)

Redução a Fecho Convexo Dinâmico

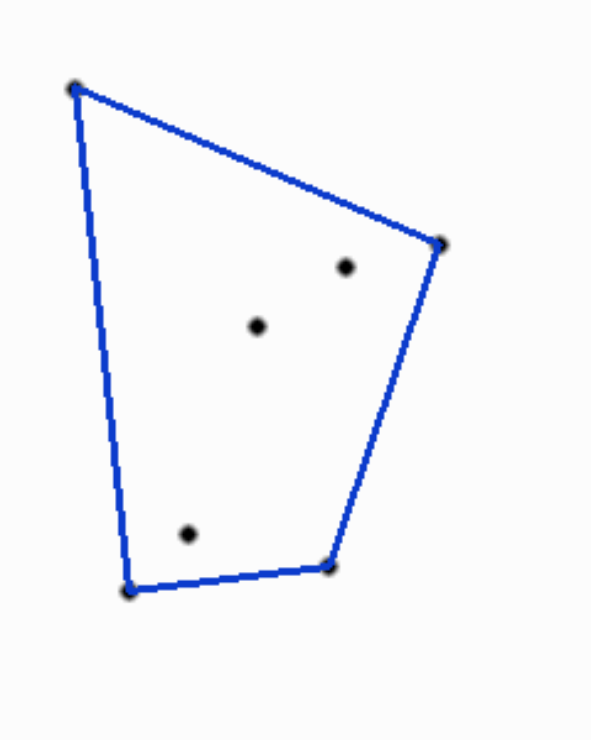
Dualidade Ponto-Reta

Ponto: (a,b)

Reta: y = ax - b



Fecho Convexo Dinâmico



FCD: O(n lg n lg lg n)
x
O(n lg^2 n α(n) / lg lg n)
(torneio e hanger (lg n)-ários)

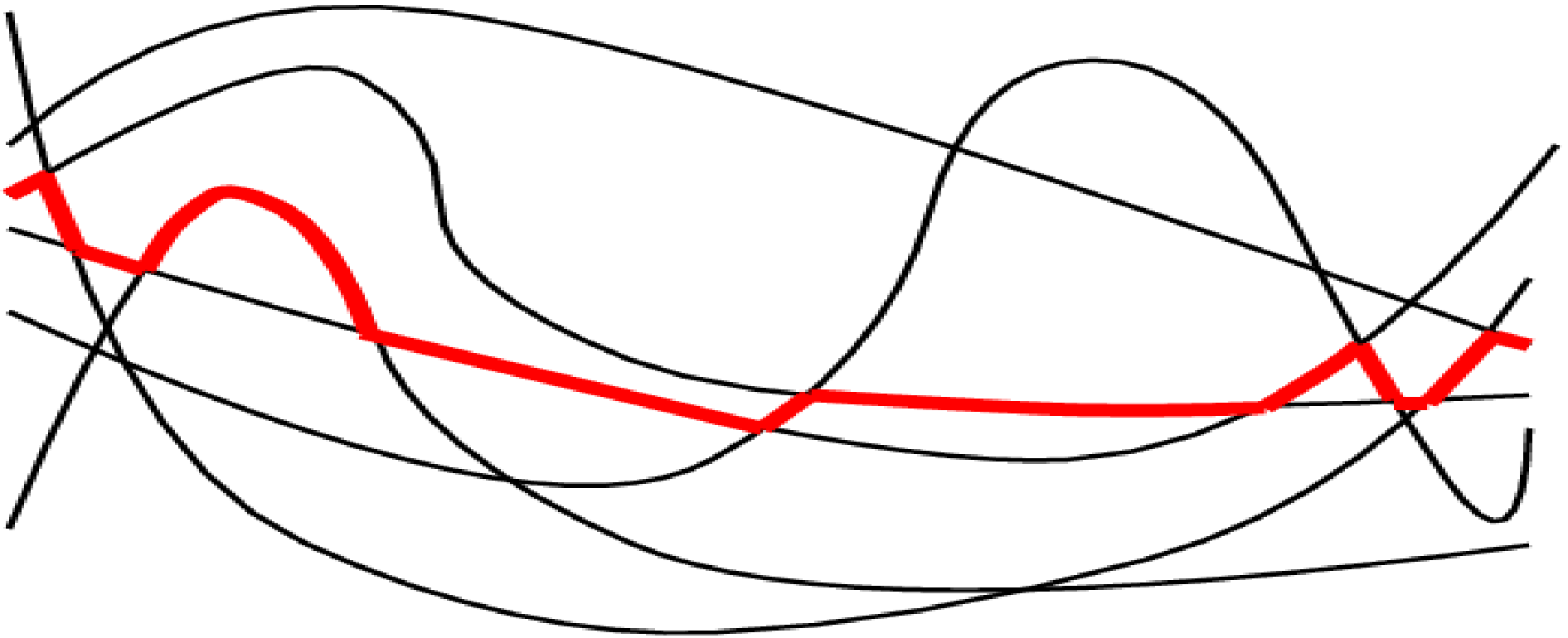
- Responde consultas como próximo ponto do fecho ou ponto extremo em uma direção em tempo O(lg n)
- Permite inserções e remoções em tempo O(lg n lg lg n)
- Lista de Prioridades Cinética mais eficiente conhecida, porém:
  - Restrita a funções lineares
  - Ineficiente na prática

Comparação das Estruturas
Aplicação
Problemas em Aberto

Comparando as Estruturas

- Eficientes: Torneio, Heater, Hanger e FCD
- Versão *k*-ária: Heap, Torneio e Hanger
- Localidade ótima: Heap, Heater e Hanger
- Determinísticas: Heap, Torneio e FCD
- Interesse prático: Heap, Torneio, Heater e Hanger
- Interesse teórico: FCD
- Restrita a retas: FCD

Aplicação: *k*-ésimo Nível



Terceiro nível de um arranjo de curvas

Problemas Abertos

- Complexidades do Heap Cinético
- Estruturas mais eficientes
- Limites inferiores
- Estrutura com complexidade semelhante a redução ao FCD, porém não restrita a retas

Muito obrigado!

Dúvidas?
Críticas?
Comentários?
Sugestões???