Análise de Algoritmos

Parte destes slides são adaptações de slides

do Prof. Paulo Feofiloff e do Prof. José Coelho de Pina.

Busca local

Problema de otimização combinatória:

conjunto de instâncias
para cada instância I,
  um conjunto $\mathrm{Sol}(I)$ (não vazio) de soluções (viáveis) e
  uma função $\mathrm{val}(S)$ para cada solução S em $\mathrm{Sol}(I)$

Busca local

Problema de otimização combinatória:

conjunto de instâncias
para cada instância I,
  um conjunto $\mathrm{Sol}(I)$ (não vazio) de soluções (viáveis) e
  uma função $\mathrm{val}(S)$ para cada solução S em $\mathrm{Sol}(I)$

Objetivo: Dada uma instância I viável,
encontrar solução S em $\mathrm{Sol}(I)$ tq $\mathrm{val}(S)$ é mínimo/máximo.

Busca local

Problema de otimização combinatória:

conjunto de instâncias
para cada instância I,
  um conjunto $\mathrm{Sol}(I)$ (não vazio) de soluções (viáveis) e
  uma função $\mathrm{val}(S)$ para cada solução S em $\mathrm{Sol}(I)$

Objetivo: Dada uma instância I viável,
encontrar solução S em $\mathrm{Sol}(I)$ tq $\mathrm{val}(S)$ é mínimo/máximo.

Tipicamente $\mathrm{Sol}(I)$ tem tamanho exponencial.

Busca local

Problema de otimização combinatória:

conjunto de instâncias
para cada instância I,
  um conjunto $\mathrm{Sol}(I)$ (não vazio) de soluções (viáveis) e
  uma função $\mathrm{val}(S)$ para cada solução S em $\mathrm{Sol}(I)$

Objetivo: Dada uma instância I viável,
encontrar solução S em $\mathrm{Sol}(I)$ tq $\mathrm{val}(S)$ é mínimo/máximo.

Tipicamente $\mathrm{Sol}(I)$ tem tamanho exponencial.

Em uma busca local, adicionamos a
noção de vizinhança entre as soluções.

Busca local

Problema de otimização combinatória:

conjunto de instâncias
para cada instância I,
  um conjunto $\mathrm{Sol}(I)$ (não vazio) de soluções (viáveis) e
  uma função $\mathrm{val}(S)$ para cada solução S em $\mathrm{Sol}(I)$

Objetivo: Dada uma instância I viável,
encontrar solução S em $\mathrm{Sol}(I)$ tq $\mathrm{val}(S)$ é mínimo/máximo.

Tipicamente $\mathrm{Sol}(I)$ tem tamanho exponencial.

Em uma busca local, adicionamos a
noção de vizinhança entre as soluções.

Relações que vizinhança simétricas.

Busca local

Algoritmo:

Começa com uma solução arbitrária S.
Iterativamente muda para uma solução S' vizinha a S.

Busca local

Algoritmo:

Começa com uma solução arbitrária S.
Iterativamente muda para uma solução S' vizinha a S.

Durante o processo, guarda a melhor solução visitada.

Busca local

Algoritmo:

Começa com uma solução arbitrária S.
Iterativamente muda para uma solução S' vizinha a S.

Durante o processo, guarda a melhor solução visitada.

Pontos críticos:

- como definir a vizinhança de uma solução

Busca local

Algoritmo:

Começa com uma solução arbitrária S.
Iterativamente muda para uma solução S' vizinha a S.

Durante o processo, guarda a melhor solução visitada.

Pontos críticos:

- como definir a vizinhança de uma solução
- como escolher S' na vizinhança de S

Busca local

Algoritmo:

Começa com uma solução arbitrária S.
Iterativamente muda para uma solução S' vizinha a S.

Durante o processo, guarda a melhor solução visitada.

Pontos críticos:

- como definir a vizinhança de uma solução
- como escolher S' na vizinhança de S

Grafo cujo conjunto de vértices é $\mathrm{Sol}(I)$ e adjacência é dada pela relação de vizinhança entre as soluções.

Algoritmo de busca local é um passeio neste grafo, que tenta se alcançar uma boa solução.

Cobertura por vértices

Grafo $G = (V, E)$

$S \subseteq V$ é cobertura por vértices se
toda aresta e em E tem pelo menos uma ponta em S.

Cobertura por vértices

Grafo $G = (V, E)$

$S \subseteq V$ é cobertura por vértices se
toda aresta e em E tem pelo menos uma ponta em S.

Problema: Dado G,
encontrar cobertura por vértices de tamanho mínimo.

custo $c(S) = |S|$

Cobertura por vértices

Grafo $G = (V, E)$

$S \subseteq V$ é cobertura por vértices se
toda aresta e em E tem pelo menos uma ponta em S.

Problema: Dado G,
encontrar cobertura por vértices de tamanho mínimo.

custo $c(S) = |S|$

Relação de vizinhança:
$S \sim S'$ se S' é obtido de S
adicionando-se ou removendo-se um vértice.

Cobertura por vértices

Grafo $G = (V, E)$

$S \subseteq V$ é cobertura por vértices se
toda aresta e em E tem pelo menos uma ponta em S.

Problema: Dado G,
encontrar cobertura por vértices de tamanho mínimo.

custo $c(S) = |S|$

Relação de vizinhança:
$S \sim S'$ se S' é obtido de S
adicionando-se ou removendo-se um vértice.

Cada solução S tem n vizinhos, onde $n = |V|$.

Algoritmo de busca local

Gradiente descendente

BUSCA-LOCAL (n, E) $\quad \triangleright\ G = (V, E),\ n = |V|$

1 $S \leftarrow [n]$

2 **enquanto** existe vizinho S' de S
tq $|S'| < |S|$ e S' é cobertura **faça**

3 $S \leftarrow S'$

4 **devolva** S

Algoritmo de busca local

Gradiente descendente

BUSCA-LOCAL (n, E) $\quad \triangleright\ G = (V, E),\ n = |V|$

1 $S \leftarrow [n]$

2 **enquanto** existe vizinho S' de S
tq $|S'| < |S|$ e S' é cobertura **faça**

3 $S \leftarrow S'$

4 **devolva** S

Exemplo 1: $G = (V, \emptyset)$

S decresce um vértice de cada vez até $S = \emptyset$, que é ótimo.

Algoritmo de busca local

Gradiente descendente

BUSCA-LOCAL (n, E) $\quad \triangleright\ G = (V, E),\ n = |V|$

1 $S \leftarrow [n]$

2 **enquanto** existe vizinho S' de S
tq $|S'| < |S|$ e S' é cobertura **faça**

3 $S \leftarrow S'$

4 **devolva** S

Exemplo 2: G é estrela de centro v

Se, ao final da primeira iteração, $S \neq S \setminus \{v\}$,
o algoritmo termina com $S = \{v\}$, que é ótimo.

Algoritmo de busca local

Gradiente descendente

BUSCA-LOCAL (n, E) $\quad \triangleright\ G = (V, E),\ n = |V|$

1 $S \leftarrow [n]$

2 **enquanto** existe vizinho S' de S
tq $|S'| < |S|$ e S' é cobertura **faça**

3 $S \leftarrow S'$

4 **devolva** S

Exemplo 2: G é estrela de centro v

Se, ao final da primeira iteração, $S = S \setminus \{v\}$,
o algoritmo termina com esse ótimo local.

Algoritmo de busca local

Gradiente descendente

BUSCA-LOCAL (n, E) $\quad \triangleright\ G = (V, E),\ n = |V|$

1 $S \leftarrow [n]$

2 **enquanto** existe vizinho S' de S
tq $|S'| < |S|$ e S' é cobertura **faça**

3 $S \leftarrow S'$

4 **devolva** S

Exemplo 2: G é estrela de centro v

O algoritmo pode se dar arbitrariamente mal.

Algoritmo Metropolis

Metropolis, Rosenbluth, Teller, e Teller (1953)

Simulação de sistema físico
de acordo com mecânica estatística.

Algoritmo Metropolis

Metropolis, Rosenbluth, Teller, e Teller (1953)

Simulação de sistema físico
de acordo com mecânica estatística.

Função de Gibbs-Boltzman: $e^{-E/(kT)}$, onde E é a energia, T é a temperatura e k é uma constante.

Algoritmo Metropolis

Metropolis, Rosenbluth, Teller, e Teller (1953)

Simulação de sistema físico
de acordo com mecânica estatística.

Função de Gibbs-Boltzman: $e^{-E/(kT)}$, onde E é a energia, T é a temperatura e k é uma constante.

Modelo: o sistema está num estado de energia E com probabilidade dada pela função de Gibbs-Boltzman.

Algoritmo Metropolis

Metropolis, Rosenbluth, Teller, e Teller (1953)

Simulação de sistema físico
de acordo com mecânica estatística.

Função de Gibbs-Boltzman: $e^{-E/(kT)}$, onde E é a energia, T é a temperatura e k é uma constante.

Modelo: o sistema está num estado de energia E com probabilidade dada pela função de Gibbs-Boltzman.

Para T fixo, a função decresce com E.

(Maior a chance de estar em um estado de menor energia.)

Algoritmo Metropolis

Metropolis, Rosenbluth, Teller, e Teller (1953)

Simulação de sistema físico
de acordo com mecânica estatística.

Função de Gibbs-Boltzman: $e^{-E/(kT)}$, onde E é a energia, T é a temperatura e k é uma constante.

Modelo: o sistema está num estado de energia E com probabilidade dada pela função de Gibbs-Boltzman.

Para T fixo, a função decresce com E.
(Maior a chance de estar em um estado de menor energia.)

Para T pequeno, estado de menor energia é mais provável que estado de energia alta.

Algoritmo Metropolis

Metropolis, Rosenbluth, Teller, e Teller (1953)

Simulação de sistema físico
de acordo com mecânica estatística.

Função de Gibbs-Boltzman: $e^{-E/(kT)}$, onde E é a energia, T é a temperatura e k é uma constante.

Modelo: o sistema está num estado de energia E com probabilidade dada pela função de Gibbs-Boltzman.

Para T fixo, a função decresce com E.
(Maior a chance de estar em um estado de menor energia.)

Para T pequeno, estado de menor energia é mais provável que estado de energia alta.

Para T grande, a diferença entre estas probabilidades é bem pequena.

Algoritmo Metropolis

Objetivo: chegar a um estado de energia mínima.

Algoritmo Metropolis

Objetivo: chegar a um estado de energia mínima.

Algoritmo Metropolis: ▷ para um dado T

Comece em um estado S.

A cada iteração, gere uma perturbação pequena S' de S.

Se $E(S') \leq E(S)$, mude para S'.

Senão seja $\Delta(E) = E(S') - E(S)$.

Mude para S' com probabilidade $e^{-\Delta(E)/(kT)}$.

Algoritmo Metropolis

Objetivo: chegar a um estado de energia mínima.

Algoritmo Metropolis: ▷ para um dado T

Comece em um estado S.

A cada iteração, gere uma perturbação pequena S' de S.

Se $E(S') \leq E(S)$, mude para S'.

Senão seja $\Delta(E) = E(S') - E(S)$.

Mude para S' com probabilidade $e^{-\Delta(E)/(kT)}$.

Seja $Z = \sum_S e^{-E(S)/(kT)}$.

Para um estado S, seja $f_S(t)$ a fração dos t primeiros passos da simulação em que o algoritmo passa em S.

$\lim_{t\to\infty} f_S(t) = \frac{1}{Z} \cdot e^{-E(S)/(kT)}$ com probabilidade indo para 1

Algoritmo Metropolis

Algoritmo Metropolis: ▷ para um dado T

Comece em um estado S.

A cada iteração, gere uma perturbação pequena S' de S.

Se $E(S') \leq E(S)$, mude para S'.

Senão seja $\Delta(E) = E(S') - E(S)$.

Mude para S' com probabilidade $e^{-\Delta(E)/(kT)}$.

Seja $Z = \sum_S e^{-E(S)/(kT)}$.

$f_S(t)$: fração dos t primeiros passos em que estamos em S.

$\lim_{t\to\infty} f_S(t) = \frac{1}{Z} \cdot e^{-E(S)/(kT)}$ com probabilidade indo para 1

Algoritmo Metropolis

Algoritmo Metropolis: ▷ para um dado T

Comece em um estado S.

A cada iteração, gere uma perturbação pequena S' de S.

Se $E(S') \leq E(S)$, mude para S'.

Senão seja $\Delta(E) = E(S') - E(S)$.

Mude para S' com probabilidade $e^{-\Delta(E)/(kT)}$.

Seja $Z = \sum_S e^{-E(S)/(kT)}$.

$f_S(t)$: fração dos t primeiros passos em que estamos em S.

$\lim_{t\to\infty} f_S(t) = \frac{1}{Z} \cdot e^{-E(S)/(kT)}$ com probabilidade indo para 1

Fica em S o tempo esperado.

Algoritmo Metropolis

Algoritmo Metropolis: ▷ para um dado T

Comece em um estado S.

A cada iteração, gere uma perturbação pequena S' de S.

Se $E(S') \leq E(S)$, mude para S'.

Senão seja $\Delta(E) = E(S') - E(S)$.

Mude para S' com probabilidade $e^{-\Delta(E)/(kT)}$.

Seja $Z = \sum_S e^{-E(S)/(kT)}$.

$f_S(t)$: fração dos t primeiros passos em que estamos em S.

$\lim_{t\to\infty} f_S(t) = \frac{1}{Z} \cdot e^{-E(S)/(kT)}$ com probabilidade indo para 1

Fica em S o tempo esperado.

Prós e contras em cima da cobertura por conjuntos.

Simulated Annealing

De volta ao sistema físico.

Annealing:
processo de cristalização em que o material é resfriado lentamente, para atingir o estado de energia mínima.

Simulated Annealing

De volta ao sistema físico.

Annealing:
processo de cristalização em que o material é resfriado lentamente, para atingir o estado de energia mínima.

Material em altas temperaturas não cristaliza.

Simulated Annealing

De volta ao sistema físico.

Annealing:
processo de cristalização em que o material é resfriado lentamente, para atingir o estado de energia mínima.

Material em altas temperaturas não cristaliza.

Se esfria rapidamente, cristaliza com muitas imperfeições.

Simulated Annealing

De volta ao sistema físico.

Annealing:
processo de cristalização em que o material é resfriado lentamente, para atingir o estado de energia mínima.

Material em altas temperaturas não cristaliza.

Se esfria rapidamente, cristaliza com muitas imperfeições.

Simulated annealing imita este processo, ajustando as probabilidades de mudar para um estado de energia maior de acordo com um parâmetro T que diminui com o tempo.

Simulated Annealing

De volta ao sistema físico.

Annealing:
processo de cristalização em que o material é resfriado lentamente, para atingir o estado de energia mínima.

Material em altas temperaturas não cristaliza.

Se esfria rapidamente, cristaliza com muitas imperfeições.

Simulated annealing imita este processo, ajustando as probabilidades de mudar para um estado de energia maior de acordo com um parâmetro T que diminui com o tempo.

Em geral, não tem garantia de qualidade.

Simulated Annealing

De volta ao sistema físico.

Annealing:
processo de cristalização em que o material é resfriado lentamente, para atingir o estado de energia mínima.

Material em altas temperaturas não cristaliza.

Se esfria rapidamente, cristaliza com muitas imperfeições.

Simulated annealing imita este processo, ajustando as probabilidades de mudar para um estado de energia maior de acordo com um parâmetro T que diminui com o tempo.

Em geral, não tem garantia de qualidade.

Em que velocidade diminuir o T?

Dois exemplos

- Redes Neurais de Hopfielg
- Problema do corte máximo