Algoritmos de Aproximação e Problemas com Seqüências

Rafael Crivellari Saliba Schouery

Orientadora: Cristina Gomes Fernandes
Instituto de Matemática e Estatística - USP

17 de novembro de 2008

Motivação para o problema

- **Um problema comum em biologia:** conseguimos seqüenciar poucas bases do DNA (de 100 a 1000).

Motivação para o problema

- **Um problema comum em biologia:** conseguimos seqüenciar poucas bases do DNA (de 100 a 1000).
- Portanto, utilizamos o seqüenciamento **shotgun**.

Motivação para o problema

- **Um problema comum em biologia:** conseguimos seqüenciar poucas bases do DNA (de 100 a 1000).
- Portanto, utilizamos o seqüenciamento **shotgun**.

AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC

Motivação para o problema

- **Um problema comum em biologia:** conseguimos seqüenciar poucas bases do DNA (de 100 a 1000).
- Portanto, utilizamos o seqüenciamento **shotgun**.

AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC

Motivação para o problema

- **Um problema comum em biologia:** conseguimos seqüenciar poucas bases do DNA (de 100 a 1000).
- Portanto, utilizamos o seqüenciamento **shotgun**.

AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC

Motivação para o problema

- **Um problema comum em biologia:** conseguimos seqüenciar poucas bases do DNA (de 100 a 1000).
- Portanto, utilizamos o seqüenciamento **shotgun**.

AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC

Motivação para o problema

- **Um problema comum em biologia:** conseguimos seqüenciar poucas bases do DNA (de 100 a 1000).
- Portanto, utilizamos o seqüenciamento **shotgun**.

AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC

Basta agora considerar apenas:

Motivação para o problema

- **Um problema comum em biologia:** conseguimos seqüenciar poucas bases do DNA (de 100 a 1000).
- Portanto, utilizamos o seqüenciamento **shotgun**.

AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC

Basta agora considerar apenas:

AGCATGCTGC, GCAGTCATGC, TGCTGCAG,
CTTAGGCTATTGG, TCATGCTTAGGCT, ATTGGACGTGATC

Motivação para o problema

- **Um problema comum em biologia:** conseguimos seqüenciar poucas bases do DNA (de 100 a 1000).
- Portanto, utilizamos o seqüenciamento **shotgun**.

AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC

Basta agora considerar apenas:

AGCATGCTGC, GCAGTCATGC, TGCTGCAG,
CTTAGGCTATTGG, TCATGCTTAGGCT, ATTGGACGTGATC

- Objetivo: reconstruir a seqüência original.

Motivação para o problema

- **Um problema comum em biologia:** conseguimos seqüenciar poucas bases do DNA (de 100 a 1000).
- Portanto, utilizamos o seqüenciamento **shotgun**.

AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC
AGCATGCTGCAGTCATGCTTAGGCTATTGGACGTGATC

Basta agora considerar apenas:

AGCATGCTGC, GCAGTCATGC, TGCTGCAG,
CTTAGGCTATTGG, TCATGCTTAGGCT, ATTGGACGTGATC

- Objetivo: reconstruir a seqüência original.
- Encontrar a menor seqüência que contenha o DNA seqüenciado.

SCS - Superseqüência comum mínima

- Para um conjunto S de seqüências desejamos encontrar uma seqüência σ de comprimento mínimo tal que todo $s \in S$ seja uma subseqüência contígua de σ.

SCS - Superseqüência comum mínima

- Para um conjunto S de seqüências desejamos encontrar uma seqüência σ de comprimento mínimo tal que todo $s \in S$ seja uma subseqüência contígua de σ.
- σ é uma **superseqüência comum mínima** para S.

SCS - Superseqüência comum mínima

- Para um conjunto S de seqüências desejamos encontrar uma seqüência σ de comprimento mínimo tal que todo $s \in S$ seja uma subseqüência contígua de σ.
- σ é uma **superseqüência comum mínima** para S.

$$S = \{\text{ATGC, AGC, GCT, CAT}\}$$

SCS - Superseqüência comum mínima

- Para um conjunto S de seqüências desejamos encontrar uma seqüência σ de comprimento mínimo tal que todo $s \in S$ seja uma subseqüência contígua de σ.
- σ é uma **superseqüência comum mínima** para S.

$$S = \{\text{ATGC, AGC, GCT, CAT}\}$$
$$\sigma = \text{AGCATGCT}$$

SCS - Superseqüência comum mínima

- Para um conjunto S de seqüências desejamos encontrar uma seqüência σ de comprimento mínimo tal que todo $s \in S$ seja uma subseqüência contígua de σ.
- σ é uma **superseqüência comum mínima** para S.

$$S = \{\text{ATGC}, \text{AGC}, \text{GCT}, \text{CAT}\}$$
$$\sigma = \text{AGCATGCT}$$

SCS - Superseqüência comum mínima

- Para um conjunto S de seqüências desejamos encontrar uma seqüência σ de comprimento mínimo tal que todo $s \in S$ seja uma subseqüência contígua de σ.
- σ é uma **superseqüência comum mínima** para S.

$$S = \{\text{ATGC, AGC, GCT, CAT}\}$$
$$\sigma = \text{AGCATGCT}$$

SCS - Superseqüência comum mínima

- Para um conjunto S de seqüências desejamos encontrar uma seqüência σ de comprimento mínimo tal que todo $s \in S$ seja uma subseqüência contígua de σ.
- σ é uma **superseqüência comum mínima** para S.

$$S = \{\text{ATGC, AGC, GCT, CAT}\}$$
$$\sigma = \text{AGCATGCT}$$

SCS - Superseqüência comum mínima

- Para um conjunto S de seqüências desejamos encontrar uma seqüência σ de comprimento mínimo tal que todo $s \in S$ seja uma subseqüência contígua de σ.
- σ é uma **superseqüência comum mínima** para S.

$$S = \{\text{ATGC, AGC, GCT, CAT}\}$$
$$\sigma = \text{AGCATGCT}$$

SCS - Superseqüência comum mínima

- Para um conjunto S de seqüências desejamos encontrar uma seqüência σ de comprimento mínimo tal que todo $s \in S$ seja uma subseqüência contígua de σ.
- σ é uma **superseqüência comum mínima** para S.

$$S = \{\text{ATGC, AGC, GCT, CAT}\}$$
$$\sigma = \text{AGCATGCT}$$

Teorema (Gallant, Maier e Storer '80)

O SCS é NP-difícil.

SCS - Superseqüência comum mínima

- Para um conjunto S de seqüências desejamos encontrar uma seqüência σ de comprimento mínimo tal que todo $s \in S$ seja uma subseqüência contígua de σ.
- σ é uma **superseqüência comum mínima** para S.

$$S = \{\text{ATGC, AGC, GCT, CAT}\}$$
$$\sigma = \text{AGCATGCT}$$

Teorema (Gallant, Maier e Storer '80)

O SCS é NP-difícil.

- Não existe algoritmo polinomial para o SCS, a menos que **P** = **NP**.

Algoritmos de Aproximação

Algoritmos de aproximação: alternativa natural para problemas difíceis.

Algoritmos de Aproximação

Algoritmos de aproximação: alternativa natural para problemas difíceis.

Duas características fundamentais:

Algoritmos de Aproximação

Algoritmos de aproximação: alternativa natural para problemas difíceis.

Duas características fundamentais:

- Eficiência

Algoritmos de Aproximação

Algoritmos de aproximação: alternativa natural para problemas difíceis.

Duas características fundamentais:

- Eficiência
- Garantia de qualidade

Algoritmos de Aproximação

Algoritmos de aproximação: alternativa natural para problemas difíceis.

Duas características fundamentais:

- Eficiência
- Garantia de qualidade

Algoritmos de Aproximação

Algoritmos de aproximação: alternativa natural para problemas difíceis.

Duas características fundamentais:

- Eficiência
- Garantia de qualidade

- Neste caso, dizemos que o algoritmo é uma α-aproximação.

Algoritmos de Aproximação

Algoritmos de aproximação: alternativa natural para problemas difíceis.
Duas características fundamentais:

- Eficiência
- Garantia de qualidade

- Neste caso, dizemos que o algoritmo é uma α-aproximação.
- Exemplo: Um algoritmo é uma **4-aproximação** para o SCS se fornece uma superseqüência comum no **máximo 4 vezes maior** do que a superseqüência comum mínima.

GREEDY

- **Algoritmos gulosos:** algoritmos que escolhem a melhor opção a cada iteração

GREEDY

- **Algoritmos gulosos:** algoritmos que escolhem a melhor opção a cada iteração
- **Uma primeira idéia:** Compomos as duas seqüências com maior sobreposição e reiteramos o processo.

GREEDY

- **Algoritmos gulosos:** algoritmos que escolhem a melhor opção a cada iteração
- **Uma primeira idéia:** Compomos as duas seqüências com maior sobreposição e reiteramos o processo.

Exemplo:

$S = \{\text{AACGTC, ACTGAA, TCTGAC, GTCACT, GTCAGG}\}$

GREEDY

- **Algoritmos gulosos:** algoritmos que escolhem a melhor opção a cada iteração
- **Uma primeira idéia:** Compomos as duas seqüências com maior sobreposição e reiteramos o processo.

Exemplo:

$S = \{$AACGTC, ACTGAA, TCTGAC, GTCACT, GTCAGG$\}$

GREEDY

- **Algoritmos gulosos:** algoritmos que escolhem a melhor opção a cada iteração
- **Uma primeira idéia:** Compomos as duas seqüências com maior sobreposição e reiteramos o processo.

Exemplo:

$$S = \{\text{AACGTC, TCTGAC, GTCACTGAA, GTCAGG}\}$$

GREEDY

- **Algoritmos gulosos:** algoritmos que escolhem a melhor opção a cada iteração
- **Uma primeira idéia:** Compomos as duas seqüências com maior sobreposição e reiteramos o processo.

Exemplo:

$$S = \{\text{AACGTC}, \text{TCTGAC}, \text{GTCACTGAA}, \text{GTCAGG}\}$$

GREEDY

- **Algoritmos gulosos:** algoritmos que escolhem a melhor opção a cada iteração
- **Uma primeira idéia:** Compomos as duas seqüências com maior sobreposição e reiteramos o processo.

Exemplo:

$$S = \{\text{AACGTCAGG}, \text{TCTGAC}, \text{GTCACTGAA}\}$$

GREEDY

- **Algoritmos gulosos:** algoritmos que escolhem a melhor opção a cada iteração
- **Uma primeira idéia:** Compomos as duas seqüências com maior sobreposição e reiteramos o processo.

Exemplo:

$$S = \{\text{TCTGACGTCACTGAACGTCAGG}\}$$

GREEDY

- **Algoritmos gulosos:** algoritmos que escolhem a melhor opção a cada iteração
- **Uma primeira idéia:** Compomos as duas seqüências com maior sobreposição e reiteramos o processo.

Exemplo:

$$S = \{\text{TCTGACGTCACTGAACGTCAGG}\}$$

Teorema (Blum, Jiang, Li, Tromp e Yannakakis '94)

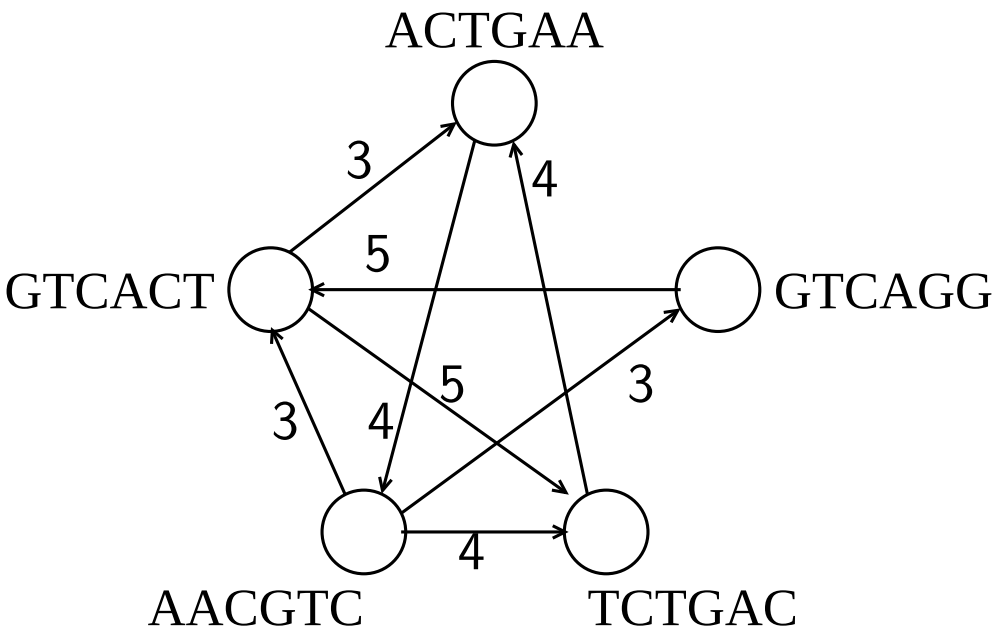
O algoritmo GREEDY é uma **4-aproximação** para o SCS.

Interpretando o GREEDY

Consideramos agora um digrafo completo que representa uma instância do problema:

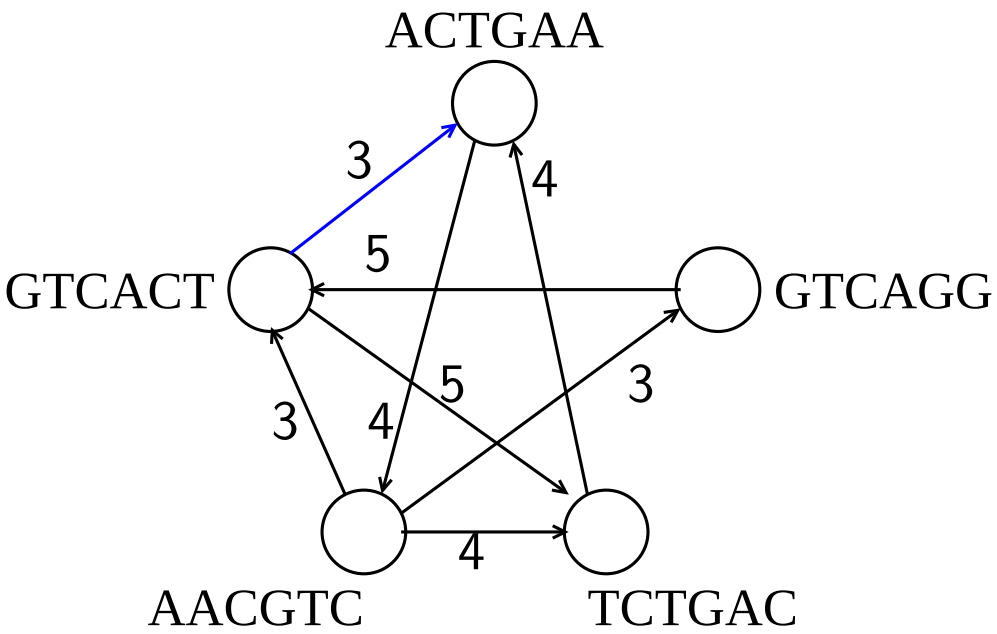
Interpretando o GREEDY

Consideramos agora um digrafo completo que representa uma instância do problema:



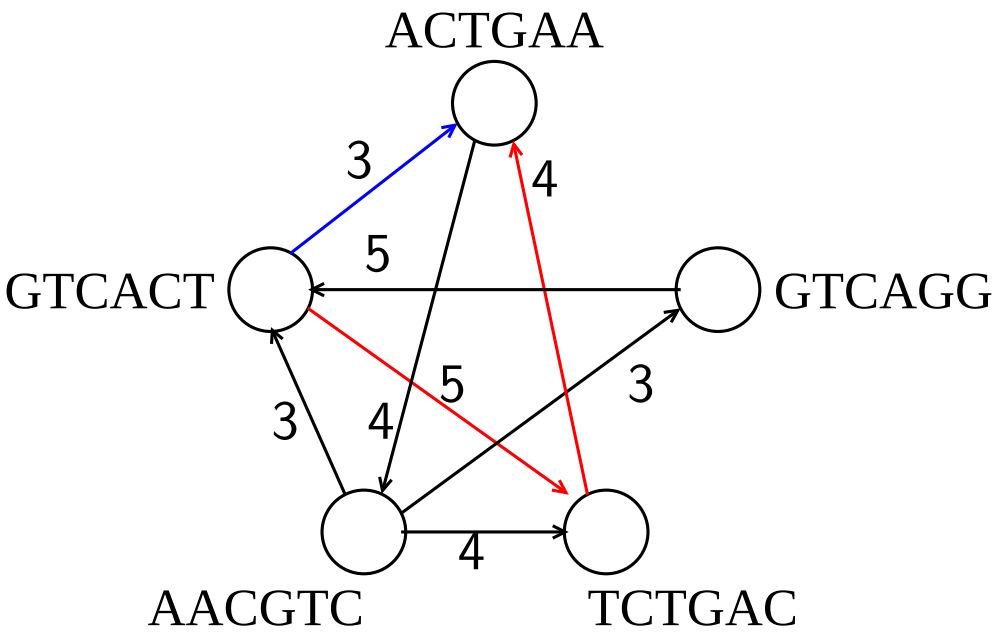
Interpretando o GREEDY

Consideramos agora um digrafo completo que representa uma instância do problema:



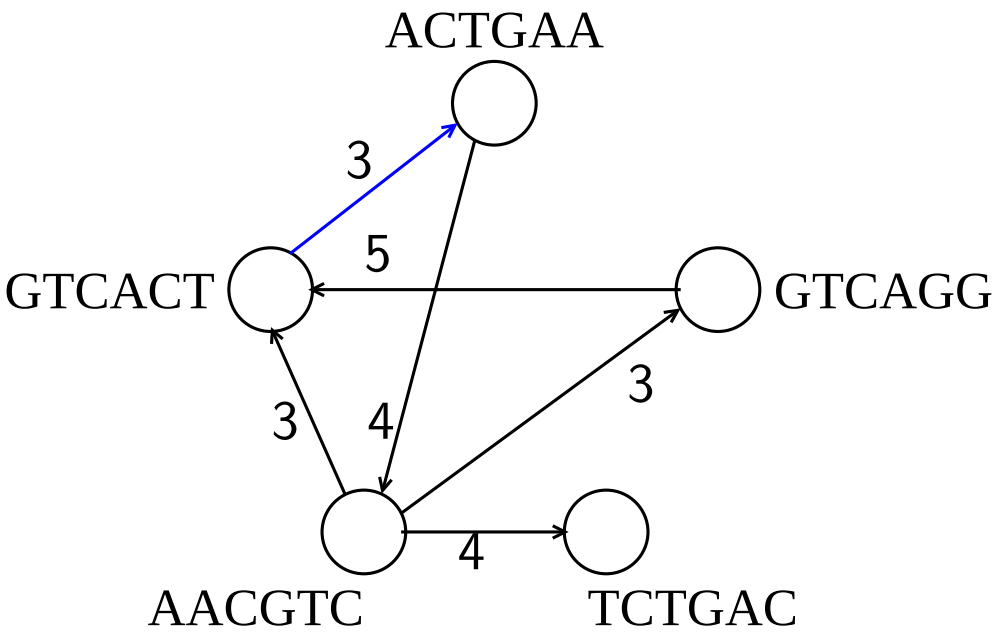
Interpretando o GREEDY

Consideramos agora um digrafo completo que representa uma instância do problema:



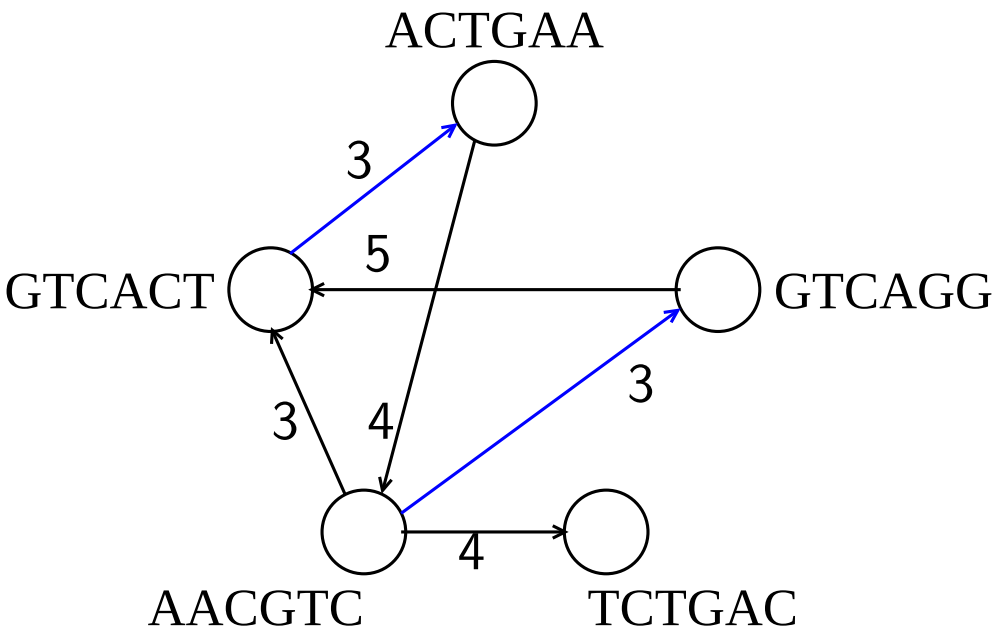
Interpretando o GREEDY

Consideramos agora um digrafo completo que representa uma instância do problema:



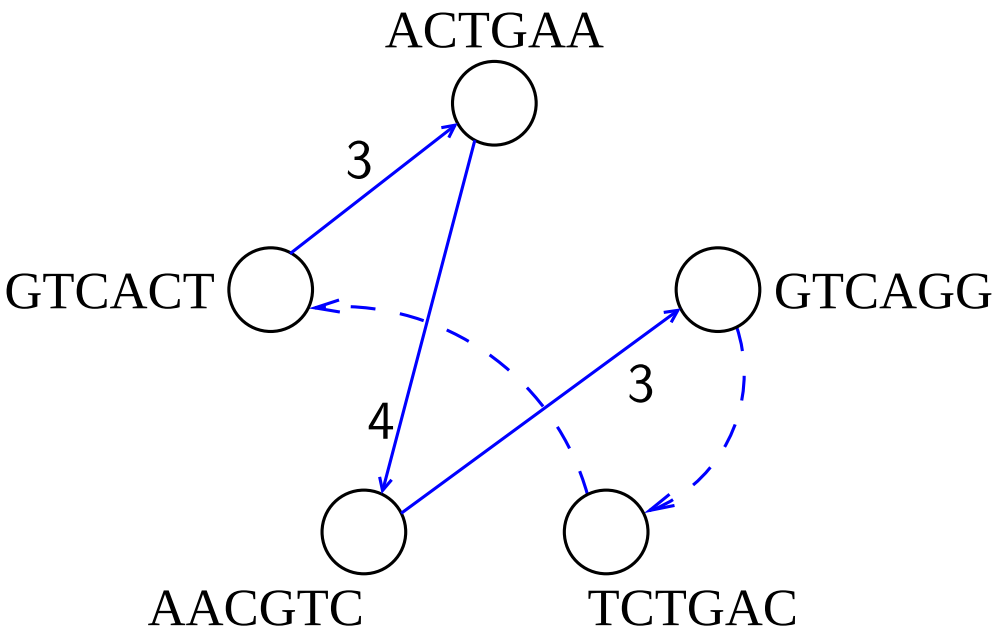
Interpretando o GREEDY

Consideramos agora um digrafo completo que representa uma instância do problema:



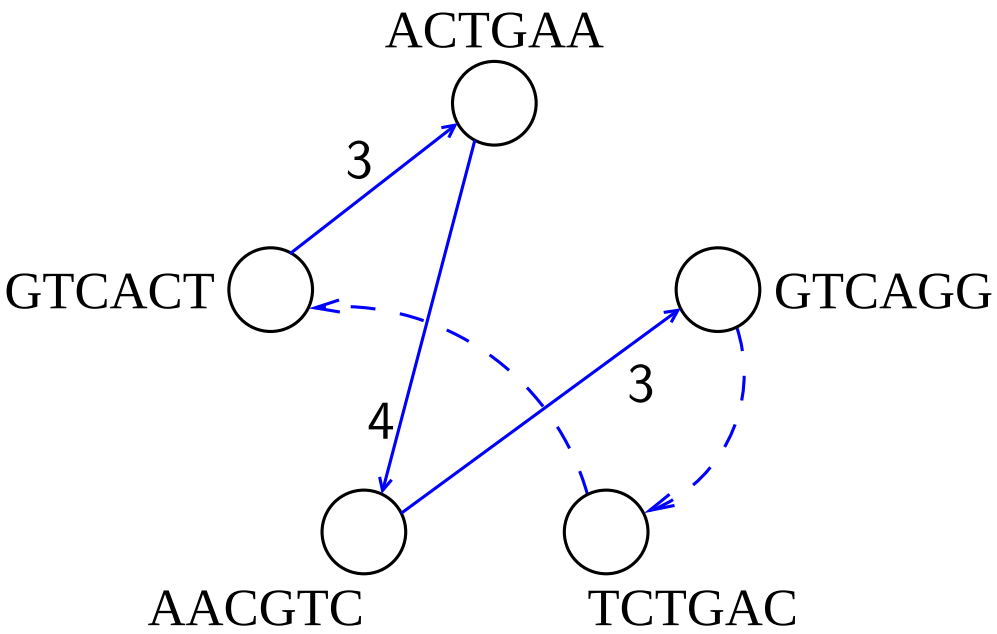
Interpretando o GREEDY

Consideramos agora um digrafo completo que representa uma instância do problema:



Interpretando o GREEDY

Consideramos agora um digrafo completo que representa uma instância do problema:



- **Custo deste circuito:** 22

Cobertura por Circuitos - MGREEDY

- TSP é NP-difícil e difícil de aproximar.

Cobertura por Circuitos - MGREEDY

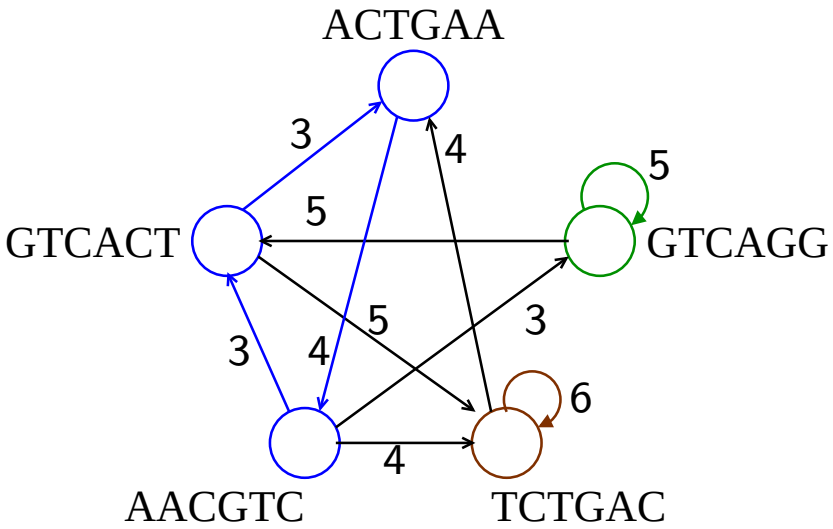
- TSP é NP-difícil e difícil de aproximar.
- Já uma cobertura por circuitos de custo mínimo que pode ser encontrada rapidamente.

Cobertura por Circuitos - MGREEDY

- TSP é NP-difícil e difícil de aproximar.
- Já uma cobertura por circuitos de custo mínimo que pode ser encontrada rapidamente.

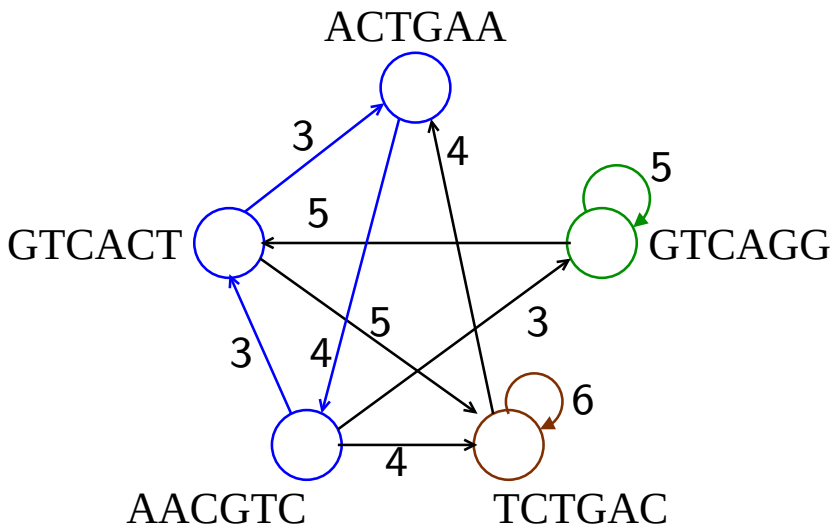
Cobertura por Circuitos - MGREEDY

- TSP é NP-difícil e difícil de aproximar.
- Já uma cobertura por circuitos de custo mínimo que pode ser encontrada rapidamente.



Cobertura por Circuitos - MGREEDY

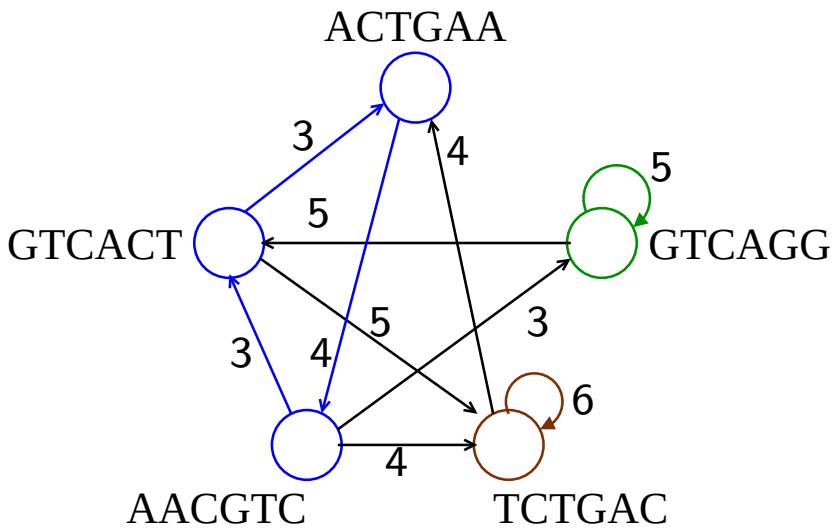
- TSP é NP-difícil e difícil de aproximar.
- Já uma cobertura por circuitos de custo mínimo que pode ser encontrada rapidamente.



- **Custo do circuito:** 21

Cobertura por Circuitos - MGREEDY

- TSP é NP-difícil e difícil de aproximar.
- Já uma cobertura por circuitos de custo mínimo que pode ser encontrada rapidamente.



- **Custo do circuito:** 21
- **Generalizando:** uma cobertura por circuitos de custo mínimo tem custo menor ou igual a de um circuito hamiltoniano.

MGREEDY

Algoritmo MGREEDY:

MGREEDY

Algoritmo MGREEDY:

- Encontramos uma cobertura por circuitos de custo mínimo.

MGREEDY

Algoritmo MGREEDY:

- Encontramos uma cobertura por circuitos de custo mínimo.

$(GTCACT, ACTGAA, AACGTC), (TCTGAC), (GTCAGG)$

MGREEDY

Algoritmo MGREEDY:

- Encontramos uma cobertura por circuitos de custo mínimo.

$(GTCACT, ACTGAA, AACGTC), (TCTGAC), (GTCAGG)$

- Encontramos as seqüências referentes aos circuitos.

MGREEDY

Algoritmo MGREEDY:

- Encontramos uma cobertura por circuitos de custo mínimo.

$(GTCACT, ACTGAA, AACGTC), (TCTGAC), (GTCAGG)$

- Encontramos as seqüências referentes aos circuitos.

$\{GTCACTGAACGTC, TCTGAC, GTCAGG\}$

MGREEDY

Algoritmo MGREEDY:

- Encontramos uma cobertura por circuitos de custo mínimo.

$(GTCACT, ACTGAA, AACGTC), (TCTGAC), (GTCAGG)$

- Encontramos as seqüências referentes aos circuitos.

$\{GTCACTGAACGTC, TCTGAC, GTCAGG\}$

- Concatenamos as seqüências encontradas.

MGREEDY

Algoritmo MGREEDY:

- Encontramos uma cobertura por circuitos de custo mínimo.

$(GTCACT, ACTGAA, AACGTC), (TCTGAC), (GTCAGG)$

- Encontramos as seqüências referentes aos circuitos.

$\{GTCACTGAACGTC, TCTGAC, GTCAGG\}$

- Concatenamos as seqüências encontradas.

$\{GTCACTGAACGTCTCTGACGTCAGG\}$

MGREEDY

Algoritmo MGREEDY:

- Encontramos uma cobertura por circuitos de custo mínimo.

$(GTCACT, ACTGAA, AACGTC), (TCTGAC), (GTCAGG)$

- Encontramos as seqüências referentes aos circuitos.

$\{GTCACTGAACGTC, TCTGAC, GTCAGG\}$

- Concatenamos as seqüências encontradas.

$\{GTCACTGAACGTCTCTGACGTCAGG\}$

Teorema (Blum, Jiang, Li, Tromp e Yannakakis '94)

O algoritmo MGREEDY é uma **4-aproximação** para o SCS.

MGREEDY

Algoritmo MGREEDY:

- Encontramos uma cobertura por circuitos de custo mínimo.

$(GTCACT, ACTGAA, AACGTC), (TCTGAC), (GTCAGG)$

- Encontramos as seqüências referentes aos circuitos.

$\{GTCACTGAACGTC, TCTGAC, GTCAGG\}$

- Concatenamos as seqüências encontradas.

$\{GTCACTGAACGTCTCTGACGTCAGG\}$

Teorema (Blum, Jiang, Li, Tromp e Yannakakis '94)

O algoritmo MGREEDY é uma **4-aproximação** para o SCS.

- E se executarmos o GREEDY nas seqüências encontradas?

MGREEDY

Algoritmo MGREEDY:

- Encontramos uma cobertura por circuitos de custo mínimo.

$(GTCACT, ACTGAA, AACGTC), (TCTGAC), (GTCAGG)$

- Encontramos as seqüências referentes aos circuitos.

$\{GTCACTGAACGTC, TCTGAC, GTCAGG\}$

- Concatenamos as seqüências encontradas.

$\{GTCACTGAACGTCTCTGACGTCAGG\}$

Teorema (Blum, Jiang, Li, Tromp e Yannakakis '94)

O algoritmo MGREEDY é uma **4-aproximação** para o SCS.

- E se executarmos o GREEDY nas seqüências encontradas?

Teorema (Blum, Jiang, Li, Tromp e Yannakakis '94)

O algoritmo TGREEDY é uma **3-aproximação** para o SCS.

Conclusão

Conjectura: O GREEDY é uma 2-aproximação para o SCS.

Conclusão

Conjectura: O GREEDY é uma 2-aproximação para o SCS.
Um exemplo de que não é melhor que uma 2-aproximação:

Conclusão

Conjectura: O GREEDY é uma 2-aproximação para o SCS.

Um exemplo de que não é melhor que uma 2-aproximação:

$S = \{c(ab)^k, (ba)^k, (ab)^k c\}$

Conclusão

Conjectura: O GREEDY é uma 2-aproximação para o SCS.
Um exemplo de que não é melhor que uma 2-aproximação:
$S = \{c(ab)^k, (ba)^k, (ab)^k c\}$

$$\frac{|c(ab)^k c(ba)^k|}{|c(ab)^{k+1} c|}$$

Conclusão

Conjectura: O GREEDY é uma 2-aproximação para o SCS.
Um exemplo de que não é melhor que uma 2-aproximação:
$S = \{c(ab)^k, (ba)^k, (ab)^k c\}$

$$\frac{|c(ab)^k c(ba)^k|}{|c(ab)^{k+1} c|} = \frac{4k+2}{2k+4}$$

Conclusão

Conjectura: O GREEDY é uma 2-aproximação para o SCS.
Um exemplo de que não é melhor que uma 2-aproximação:
$S = \{c(ab)^k, (ba)^k, (ab)^k c\}$

$$\frac{|c(ab)^k c(ba)^k|}{|c(ab)^{k+1} c|} = \frac{4k+2}{2k+4} \approx 2$$

Conclusão

Conjectura: O GREEDY é uma 2-aproximação para o SCS.
Um exemplo de que não é melhor que uma 2-aproximação:
$S = \{c(ab)^k, (ba)^k, (ab)^k c\}$

$$\frac{|c(ab)^k c(ba)^k|}{|c(ab)^{k+1} c|} = \frac{4k+2}{2k+4} \approx 2$$

Teorema (Kaplan e Shafrir '05)

O algoritmo GREEDY é uma $3\frac{1}{2}$-aproximação para o SCS.

Conclusão

Conjectura: O GREEDY é uma 2-aproximação para o SCS.
Um exemplo de que não é melhor que uma 2-aproximação:
$S = \{c(ab)^k, (ba)^k, (ab)^k c\}$

$$\frac{|c(ab)^k c(ba)^k|}{|c(ab)^{k+1} c|} = \frac{4k+2}{2k+4} \approx 2$$

Teorema (Kaplan e Shafrir '05)

O algoritmo GREEDY é uma $3\frac{1}{2}$-aproximação para o SCS.

Teorema (Sweedyk, '98)

Existe uma $2\frac{1}{2}$-aproximação para o SCS.

Conclusão

Conjectura: O GREEDY é uma 2-aproximação para o SCS.
Um exemplo de que não é melhor que uma 2-aproximação:
$S = \{c(ab)^k, (ba)^k, (ab)^k c\}$

$$\frac{|c(ab)^k c(ba)^k|}{|c(ab)^{k+1} c|} = \frac{4k+2}{2k+4} \approx 2$$

Teorema (Kaplan e Shafrir '05)

O algoritmo GREEDY é uma $3\frac{1}{2}$-aproximação para o SCS.

Teorema (Sweedyk, '98)

Existe uma $2\frac{1}{2}$-aproximação para o SCS.

Os resultados mostram a importância da conjectura.

Conclusão

Conjectura: O GREEDY é uma 2-aproximação para o SCS.
Um exemplo de que não é melhor que uma 2-aproximação:
$S = \{c(ab)^k, (ba)^k, (ab)^k c\}$

$$\frac{|c(ab)^k c(ba)^k|}{|c(ab)^{k+1} c|} = \frac{4k+2}{2k+4} \approx 2$$

Teorema (Kaplan e Shafrir '05)

O algoritmo GREEDY é uma $3\frac{1}{2}$-aproximação para o SCS.

Teorema (Sweedyk, '98)

Existe uma $2\frac{1}{2}$-aproximação para o SCS.

Os resultados mostram a importância da conjectura.

- O GREEDY é simples e rápido.

Conclusão

Conjectura: O GREEDY é uma 2-aproximação para o SCS.
Um exemplo de que não é melhor que uma 2-aproximação:
$S = \{c(ab)^k, (ba)^k, (ab)^k c\}$

$$\frac{|c(ab)^k c(ba)^k|}{|c(ab)^{k+1} c|} = \frac{4k+2}{2k+4} \approx 2$$

Teorema (Kaplan e Shafrir '05)

O algoritmo GREEDY é uma $3\frac{1}{2}$-aproximação para o SCS.

Teorema (Sweedyk, '98)

Existe uma $2\frac{1}{2}$-aproximação para o SCS.

Os resultados mostram a importância da conjectura.

- O GREEDY é simples e rápido.
- O GREEDY seria a melhor aproximação conhecida para o SCS.

Obrigado!

Rafael Crivellari Saliba Schouery
Orientadora: Cristina Gomes Fernandes
Departamento de Ciência da Computação
Instituto de Matemática e Estatística - USP
schouery@ime.usp.br

A monografia está disponível em:
http://www.ime.usp.br/∼schouery/scs/monografia.pdf