Greedy conjecture for strings of length 4

A. Kulikov, S. Savinov, E. Sluzhaev

CPM 2015

Part I

Shortest common superstring: overview

Problem statement

- The shortest common superstring problem (SCS) is: given a set $\{s_1, \ldots, s_n\}$ of n strings find a shortest string containing each s_i as a substring.

Problem statement

- The shortest common superstring problem (SCS) is: given a set $\{s_1, \ldots, s_n\}$ of n strings find a shortest string containing each s_i as a substring.
- Practical applications: data storage, data compression, genome assembly.

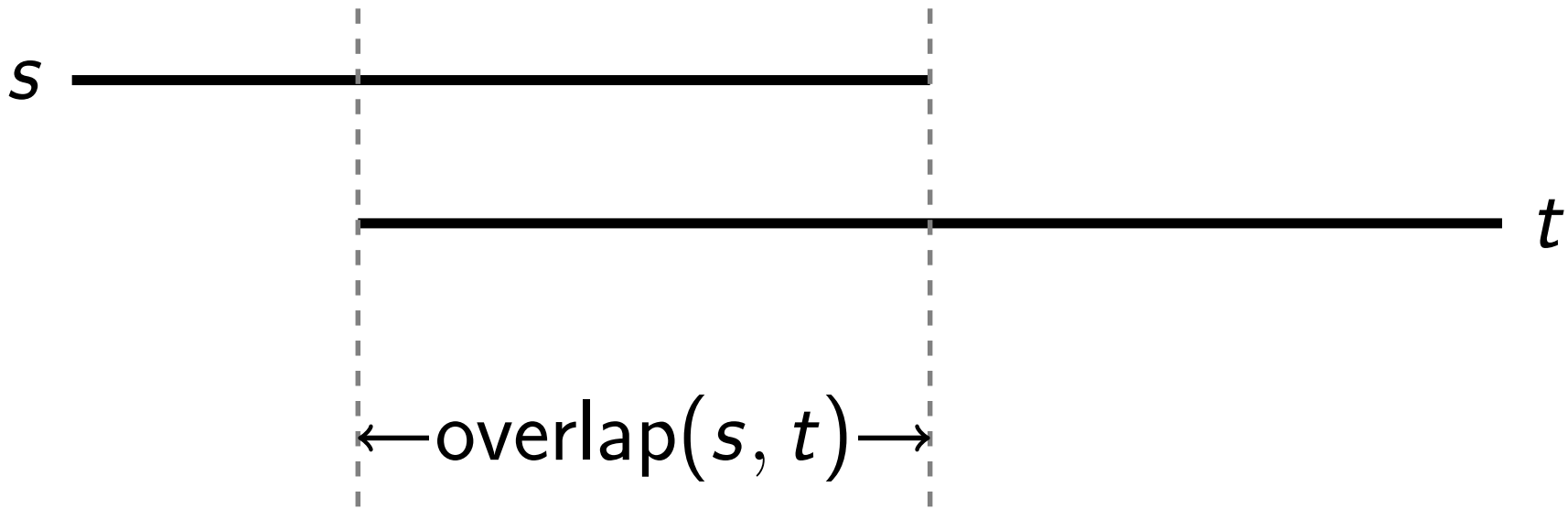
Known results

- Exact: 2^n
  - Dynamic programming, exp space [Bellman, 1960; Held, Karp, 1962]
  - Inclusion-exclusion, poly space [Kohn, Gottlieb, Kohn, 1977; Karp, 1982; Bax, 1993]
  - $2^{n-\Omega(\sqrt{n/\log n})}$ [Björklund, 2012]
- Approximation ratio: $2\frac{11}{23}$ [Mucha, 2013]
- Inapproximability ratio: $1\frac{1}{332}$ [Karpinski and Schmied, 2013]

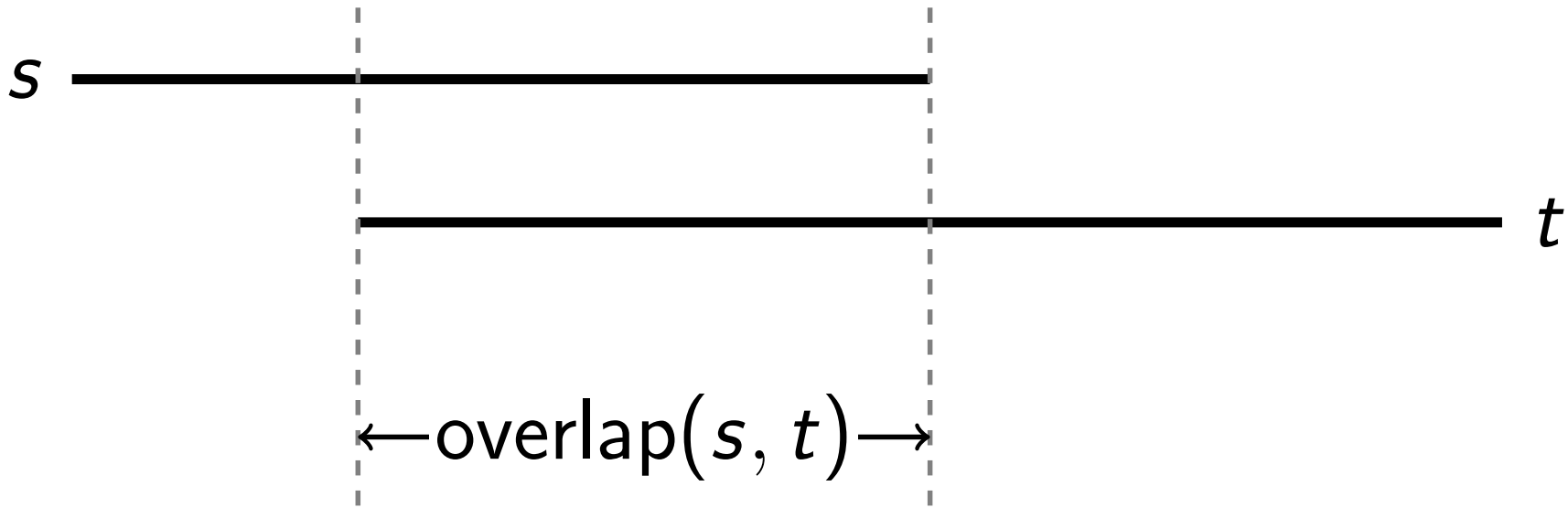
Known results for special cases

- r-SCS: all input strings have length r.
- 2-SCS $\in$ P, while for $r \geq 3$, r-SCS is NP-hard [Gallant et al., 1980].
- SCS over $\{0, 1\}$ is NP-hard [Gallant et al., 1980].

Overlap



Overlap

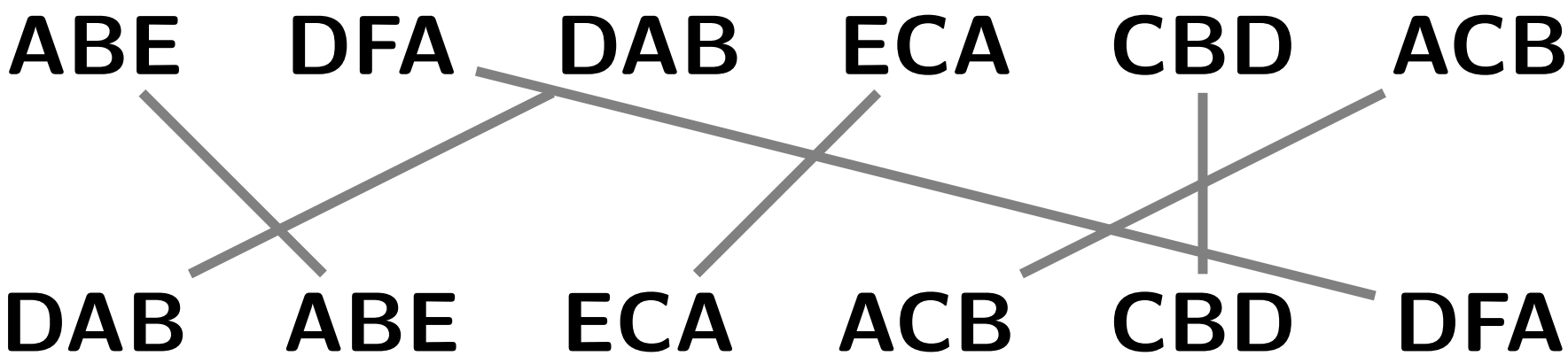


$$|\,\mathrm{overlap}(\mathrm{DAB}, \mathrm{ABE})| = 2$$
$$|\,\mathrm{overlap}(\mathrm{DAB}, \mathrm{ADE})| = 0$$

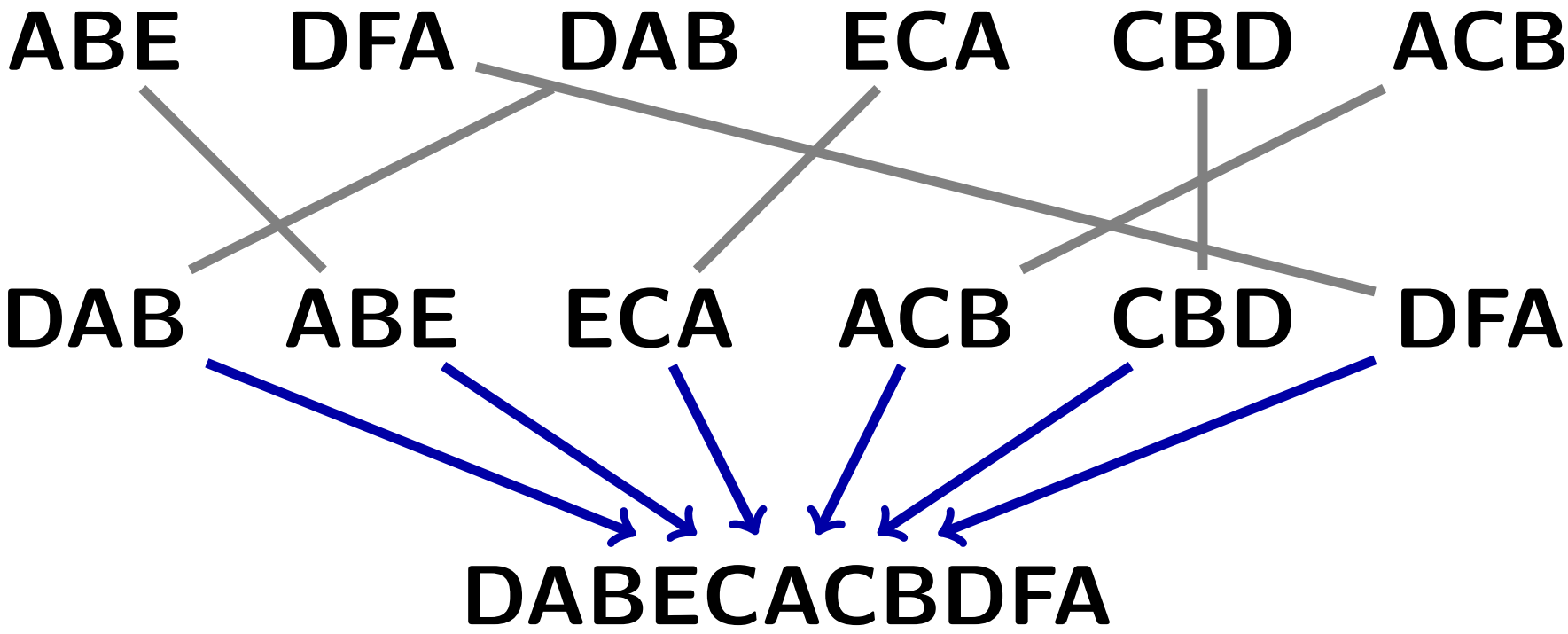
SCS: permutation problem

ABE DFA DAB ECA CBD ACB

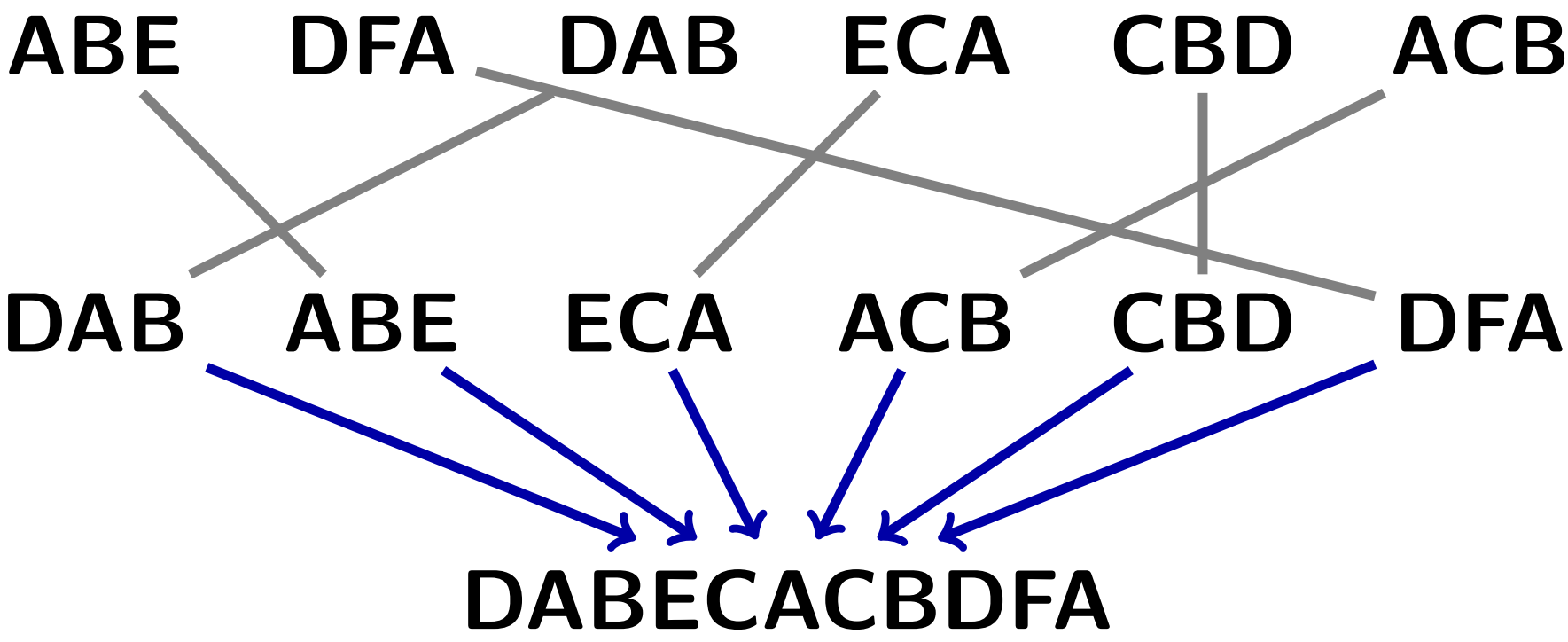
SCS: permutation problem



SCS: permutation problem



SCS: permutation problem



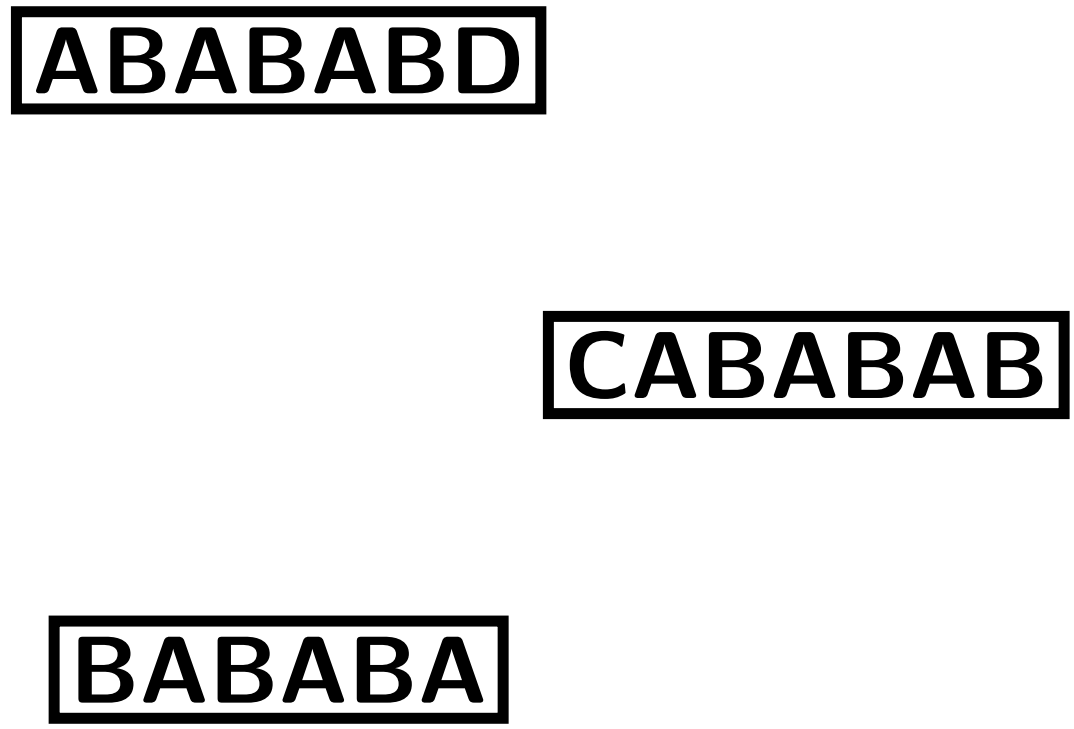
Length of superstring is $\sum_i |s_i| - c^{opt}$

Overlap graph

$$S = \{CABABAB, ABABABD, BABABA\}$$

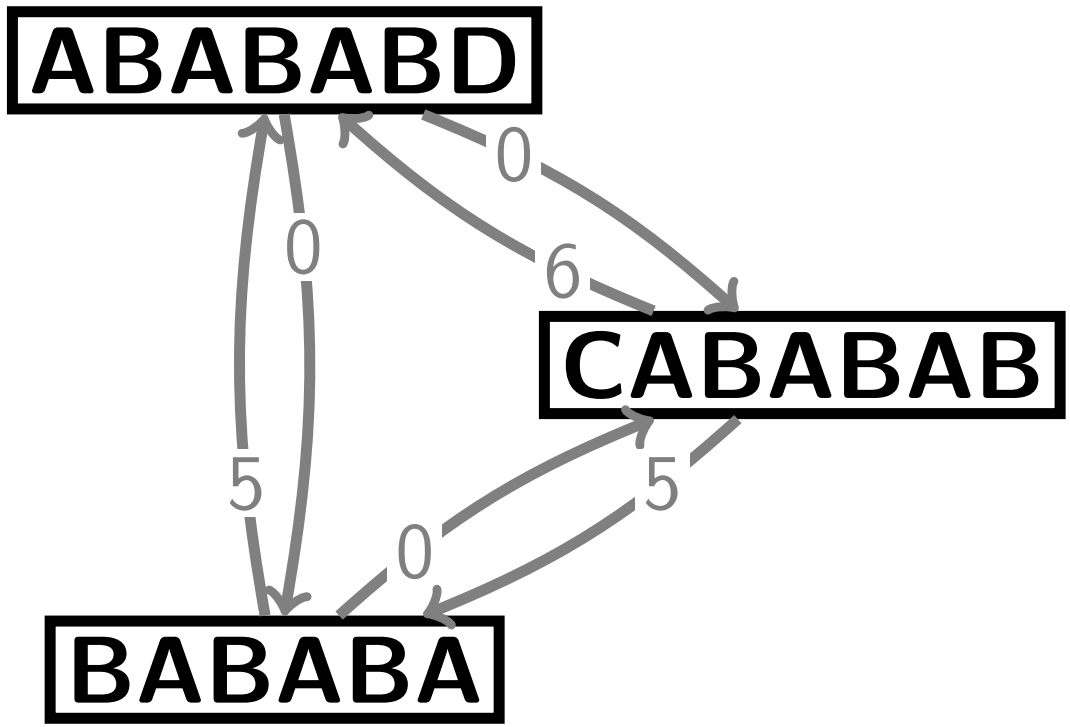
Overlap graph

$$S = \{CABABAB, ABABABD, BABABA\}$$



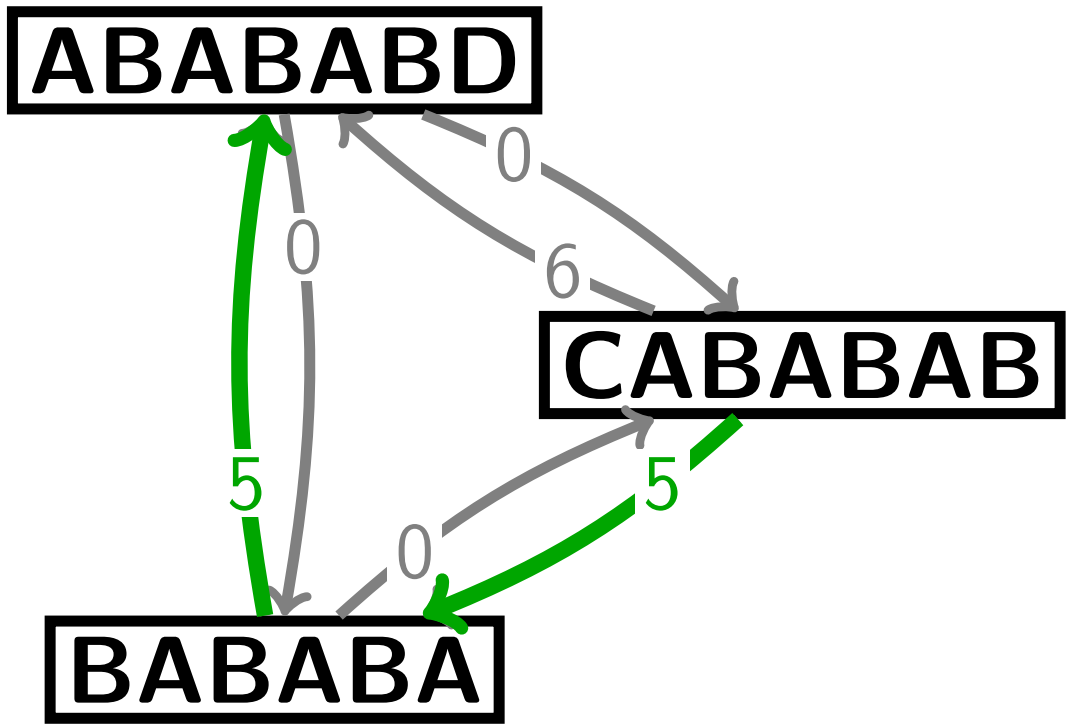
Overlap graph

$S = \{CABABAB, ABABABD, BABABA\}$



Overlap graph

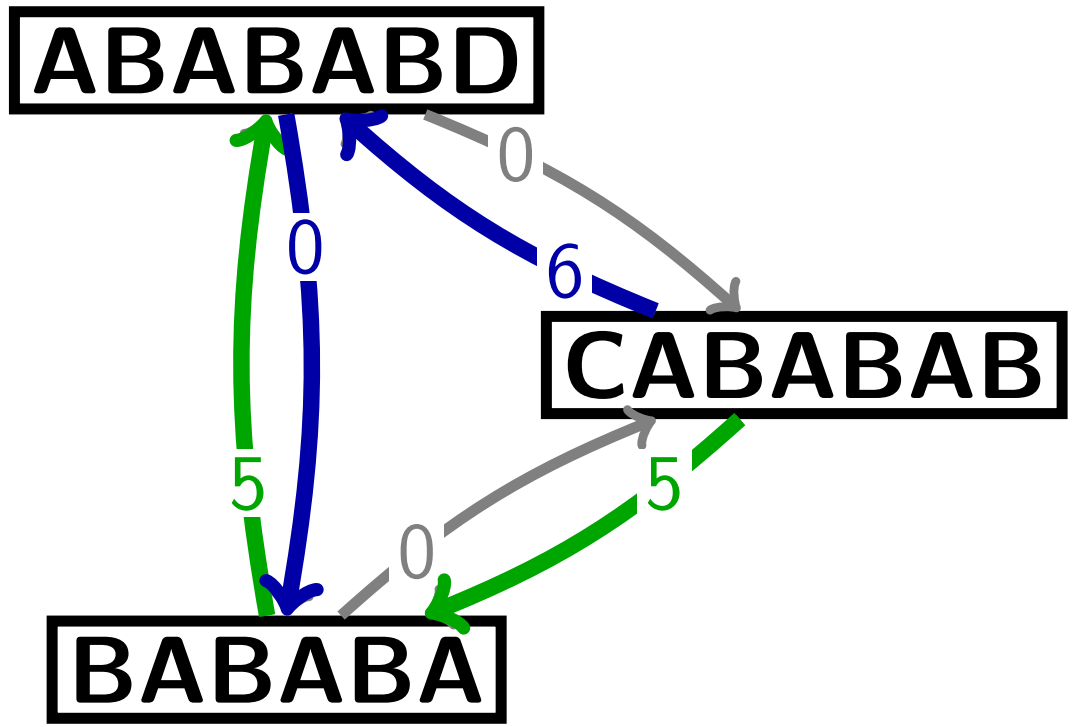
$S = \{CABABAB, ABABABD, BABABA\}$



OPT: CABABABABD (10), $c^{opt} = 10$

Overlap graph

$S = \{CABABAB, ABABABD, BABABA\}$



OPT: CABABABABD (10), $c^{opt} = 10$

GREEDY: CABABABDBABABA (14), $c^{gr} = 6$

Greedy conjecture

A superstring resulting from such a greedy process is at most two times longer than an optimal superstring. [Gallant, 1982; Turner, 1989]

Greedy conjecture in terms of compression

- Greedy conjecture for r-SCS is equivalent to $rn - c^{gr} \leq 2(rn - c^{opt})$ or $2c^{opt} \leq rn + c^{gr}$

Greedy conjecture in terms of compression

- Greedy conjecture for r-SCS is equivalent to $rn - c^{gr} \leq 2(rn - c^{opt})$ or $2c^{opt} \leq rn + c^{gr}$
- Greedy conjecture for 3-SCS is equivalent to $2c^{opt} \leq 3n + c^{gr}$

Greedy conjecture in terms of compression

- Greedy conjecture for r-SCS is equivalent to $rn - c^{gr} \leq 2(rn - c^{opt})$ or $2c^{opt} \leq rn + c^{gr}$
- Greedy conjecture for 3-SCS is equivalent to $2c^{opt} \leq 3n + c^{gr}$
- $c^{opt} \leq 2n$ is true for 3-strings

Greedy conjecture in terms of compression

- Greedy conjecture for r-SCS is equivalent to $rn - c^{gr} \leq 2(rn - c^{opt})$ or $2c^{opt} \leq rn + c^{gr}$
- Greedy conjecture for 3-SCS is equivalent to $2c^{opt} \leq 3n + c^{gr}$
- $c^{opt} \leq 2n$ is true for 3-strings
- It is sufficient to show that $0.5c^{opt} \leq c^{gr}$

Greedy conjecture in terms of compression

- Greedy conjecture for r-SCS is equivalent to $rn - c^{gr} \leq 2(rn - c^{opt})$ or $2c^{opt} \leq rn + c^{gr}$
- Greedy conjecture for 3-SCS is equivalent to $2c^{opt} \leq 3n + c^{gr}$
- $c^{opt} \leq 2n$ is true for 3-strings
- It is sufficient to show that $0.5c^{opt} \leq c^{gr}$
- It is 2-approximation of compression [Tarhio and Ukkonen, 1986]

Part II

Our results

Case of 4-strings (1)

- The greedy conjecture for 3-strings follows from the fact that the greedy algorithm approximates the compression within a factor of 2

Case of 4-strings (1)

- The greedy conjecture for 3-strings follows from the fact that the greedy algorithm approximates the compression within a factor of 2
- This approach does not work for 4-strings

Case of 4-strings (1)

- The greedy conjecture for 3-strings follows from the fact that the greedy algorithm approximates the compression within a factor of 2
- This approach does not work for 4-strings
- Thus, to prove the greedy conjecture for 4-strings, we consider overlaps of different length separately

Case of 4-strings (2)

- Let $\#_i$ be the number of overlaps of length i

Case of 4-strings (2)

- Let $\#_i$ be the number of overlaps of length i
- Compression for 4-strings is $\#_1 + 2\#_2 + 3\#_3$

Case of 4-strings (2)

- Let $\#_i$ be the number of overlaps of length i
- Compression for 4-strings is $\#_1 + 2\#_2 + 3\#_3$
- Greedy conjecture is equivalent to $2\#_1^{\text{opt}} + 4\#_2^{\text{opt}} + 6\#_3^{\text{opt}} \leq 4n + \#_1^{\text{gr}} + 2\#_2^{\text{gr}} + 3\#_3^{\text{gr}}$

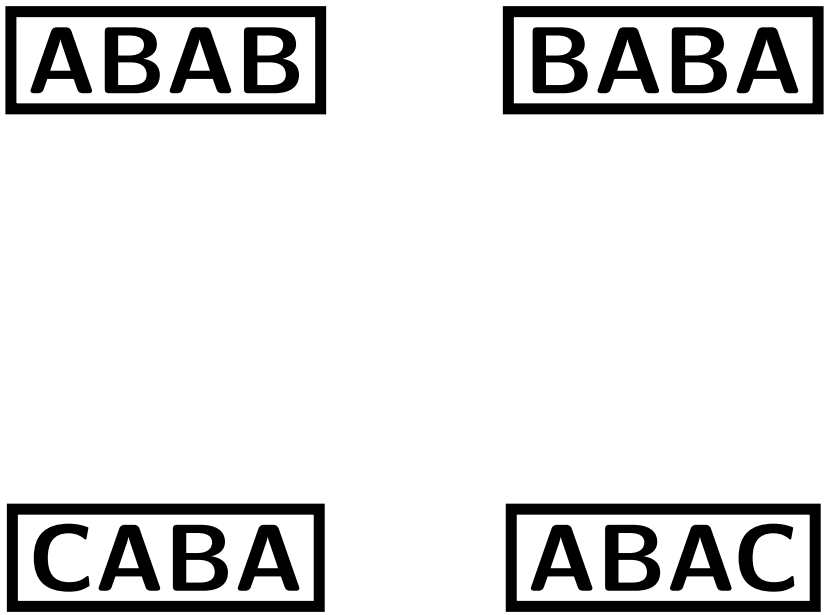
Case of 4-strings (2)

- Let $\#_i$ be the number of overlaps of length i
- Compression for 4-strings is $\#_1 + 2\#_2 + 3\#_3$
- Greedy conjecture is equivalent to $2\#_1^{\text{opt}} + 4\#_2^{\text{opt}} + 6\#_3^{\text{opt}} \leq 4n + \#_1^{\text{gr}} + 2\#_2^{\text{gr}} + 3\#_3^{\text{gr}}$
- Clearly, $\#_1^{\text{opt}} + \#_2^{\text{opt}} + \#_3^{\text{opt}} \leq n$

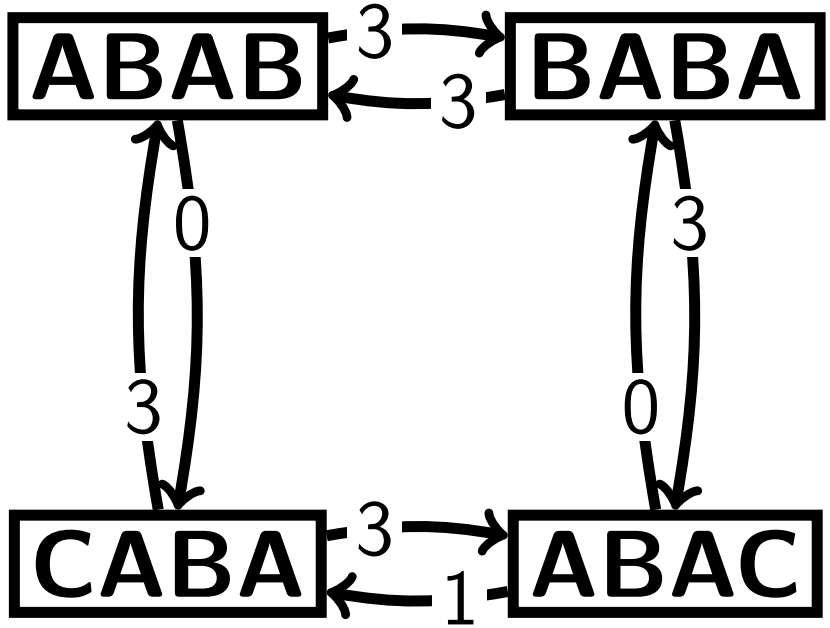
Case of 4-strings (2)

- Let $\#_i$ be the number of overlaps of length i
- Compression for 4-strings is $\#_1 + 2\#_2 + 3\#_3$
- Greedy conjecture is equivalent to $2\#_1^{\text{opt}} + 4\#_2^{\text{opt}} + 6\#_3^{\text{opt}} \leq 4n + \#_1^{\text{gr}} + 2\#_2^{\text{gr}} + 3\#_3^{\text{gr}}$
- Clearly, $\#_1^{\text{opt}} + \#_2^{\text{opt}} + \#_3^{\text{opt}} \leq n$
- It is enough to show that $2\#_3^{\text{opt}} \leq 3\#_3^{\text{gr}} + 2\#_2^{\text{gr}} + \#_1^{\text{gr}}$

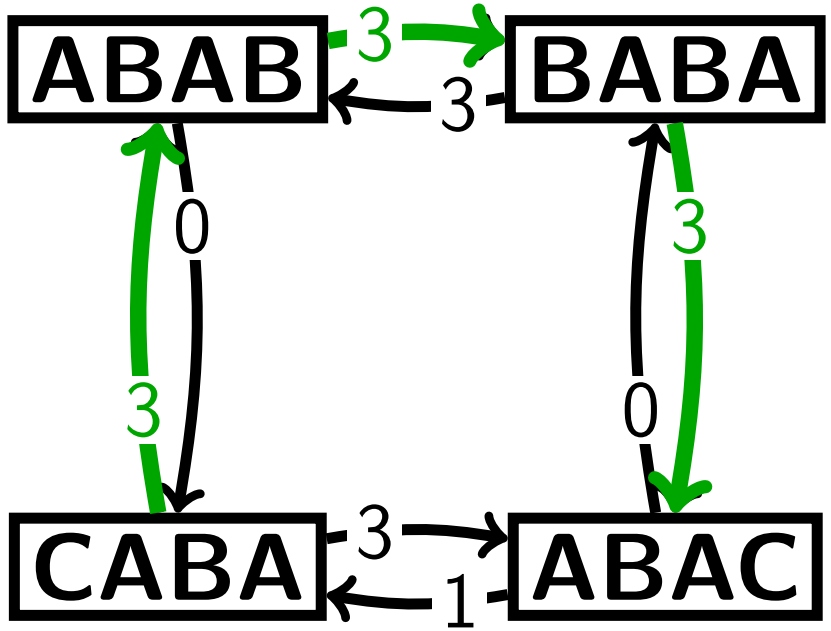
Analysis of greedy algorithm



Analysis of greedy algorithm

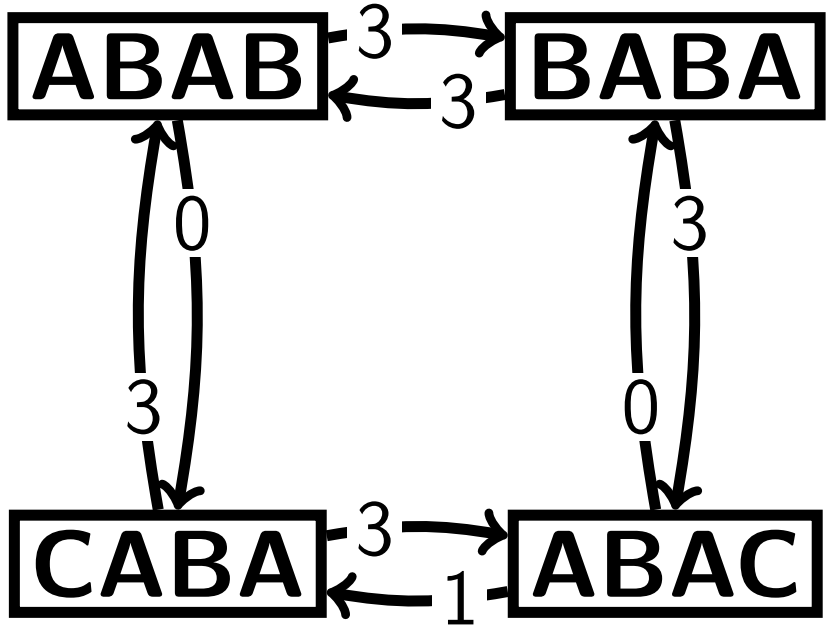


Analysis of greedy algorithm



OPT: CABABAC (7), $c^{opt} = 9$

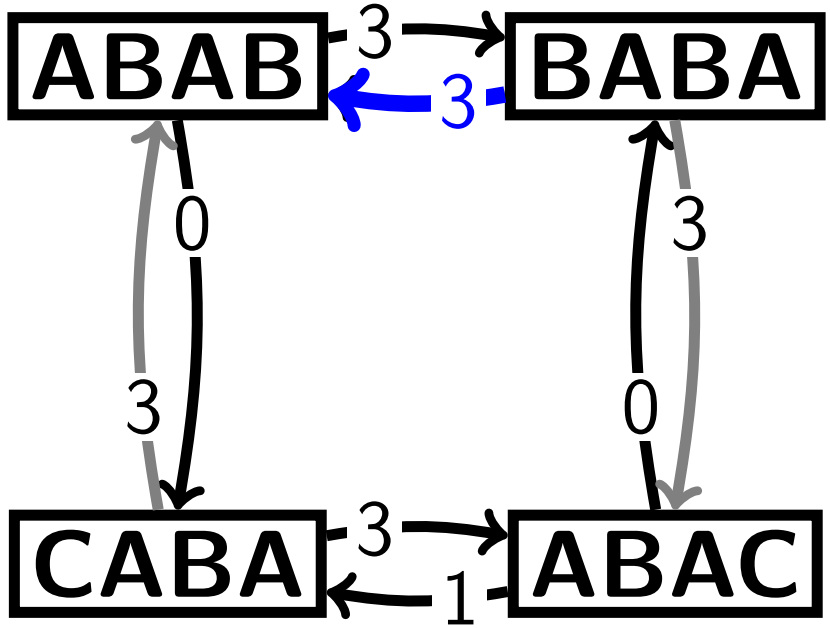
Analysis of greedy algorithm



OPT: CABABAC (7), $c^{opt} = 9$

GREEDY:

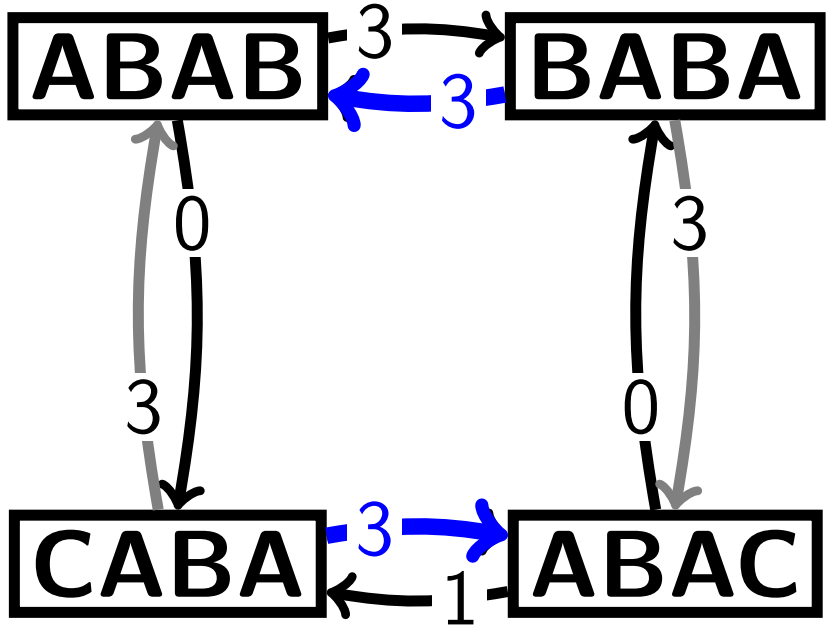
Analysis of greedy algorithm



OPT: CABABAC (7), $c^{opt} = 9$

GREEDY:

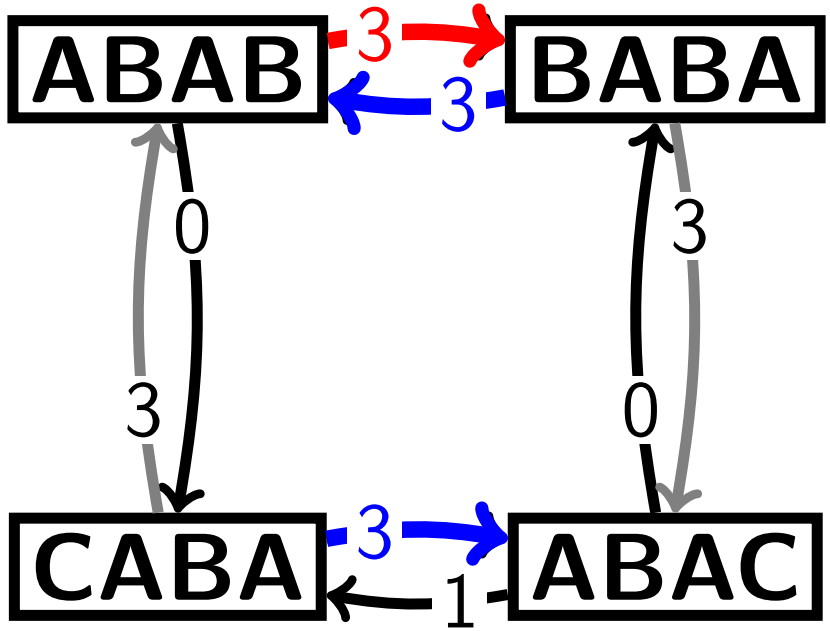
Analysis of greedy algorithm



OPT: CABABAC (7), $c^{opt} = 9$

GREEDY:

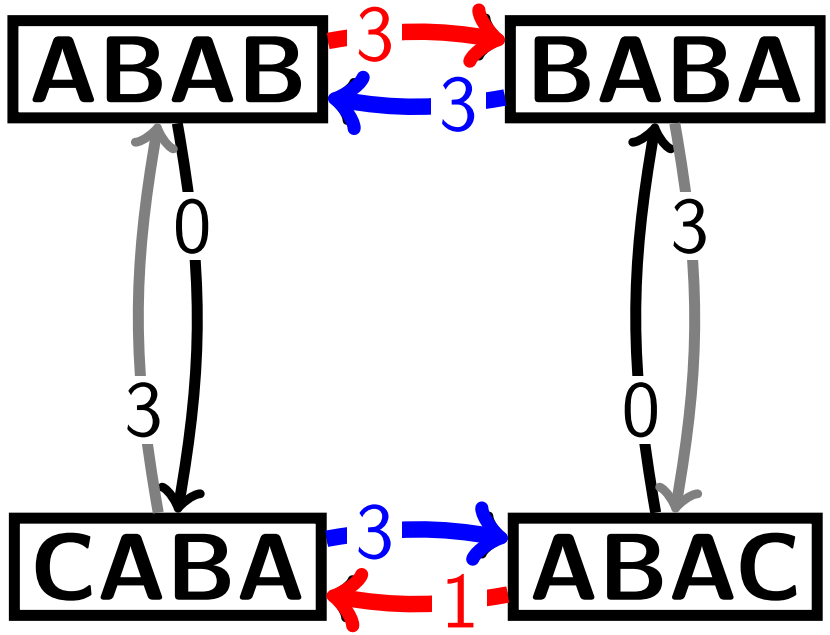
Analysis of greedy algorithm



OPT: CABABAC (7), $c^{opt} = 9$

GREEDY:

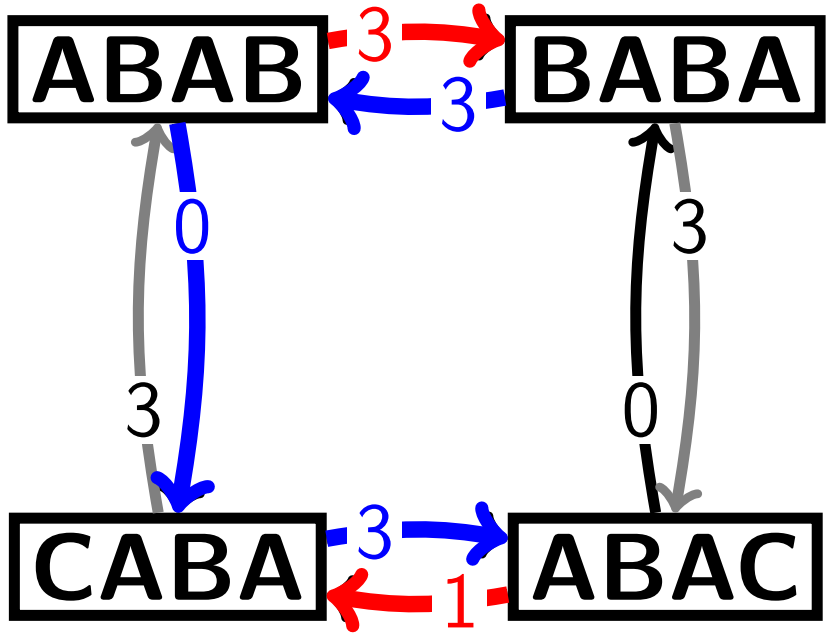
Analysis of greedy algorithm



OPT: CABABAC (7), $c^{opt} = 9$

GREEDY:

Analysis of greedy algorithm



OPT: CABABAC (7), $c^{opt} = 9$

GREEDY: ABABACABAC (10), $c^{gr} = 6$

Observations

- The greedy algorithm finds an optimum cycle cover in the overlap graph [Blum et al., 1994]

Observations

- The greedy algorithm finds an optimum cycle cover in the overlap graph [Blum et al., 1994]
- The only case when the algorithm cannot take an edge is when it forms a cycle. If at each iteration the algorithm takes an edge of maximum weight then the resulting superstring is optimal.

Proof for 4-strings (1)

- Recall that it is enough to show that $2\#_3^{\text{opt}} \leq 3\#_3^{\text{gr}} + 2\#_2^{\text{gr}} + \#_1^{\text{gr}}$

Proof for 4-strings (1)

- Recall that it is enough to show that $2\#_3^{\text{opt}} \leq 3\#_3^{\text{gr}} + 2\#_2^{\text{gr}} + \#_1^{\text{gr}}$
- Bad situation is when we have a cycle formed by overlaps of length 3

Proof for 4-strings (1)

- Recall that it is enough to show that $2\#_3^{\text{opt}} \leq 3\#_3^{\text{gr}} + 2\#_2^{\text{gr}} + \#_1^{\text{gr}}$
- Bad situation is when we have a cycle formed by overlaps of length 3
- In case of such cycles with length at least 3 we have a good upper bound for its number: $0.5\#_3^{\text{gr}}$ (due to fact that cycle without one edge contains at least 2 overlaps)

Proof for 4-strings (2)

- The worst case is 2-cycle which has form *abab* - *baba*

Proof for 4-strings (2)

- The worst case is 2-cycle which has form *abab* - *baba*
- We have analyzed carefully this case and have provided good upper bound for number of such cycles

Thank you for your attention!