Further Improvement in Approximating the Maximum Duo-Preservation String Mapping Problem

Brian Brubach

University of Maryland, College Park

16th Workshop on Algorithms in Bioinformatics – WABI 2016

Nicknames

- Maximum Duo-preservation String Mapping problem (MPSM)
  - Duo-preservation problem
- Minimum Common String Partition problem (MCSP)
  - String Partition problem
- The aroncyms for tehse plorebms are aslo hrad to tlel arapt
  - Yuor bairn can slove tshee pbroelms

Motivations

- Comparing biological sequences which differ due to some process of rearrangements
- *A Linear Time Approximation Algorithm for the DCJ Distance for Genomes with Bounded Number of Duplicates*
  - Rubert, Feijão, Braga, Stoye, Martinez (WABI '16)
  - Utilizes an approximation algorithm for MCSP
- Data compression of similar strings using rearrangements

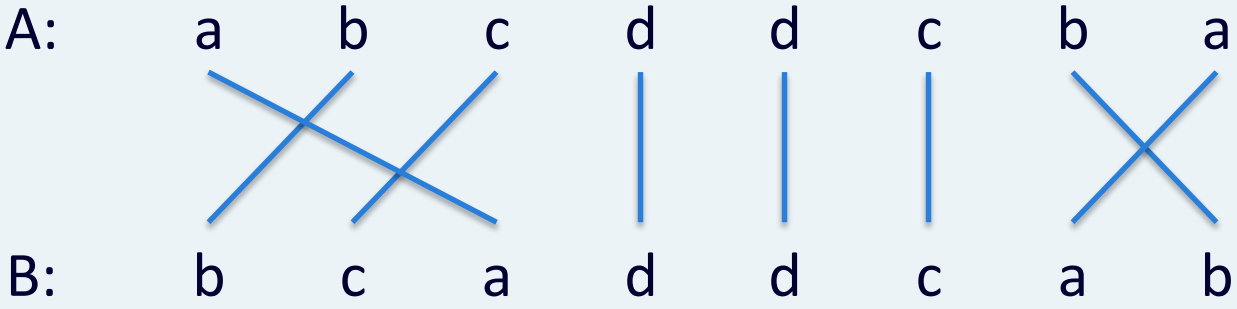
Max Duo-preservation String Mapping Problem (MPSM)

- Input: two strings which are permutations of each other

A:	a	b	c	d	d	c	b	a
B:	b	c	a	d	d	c	a	b

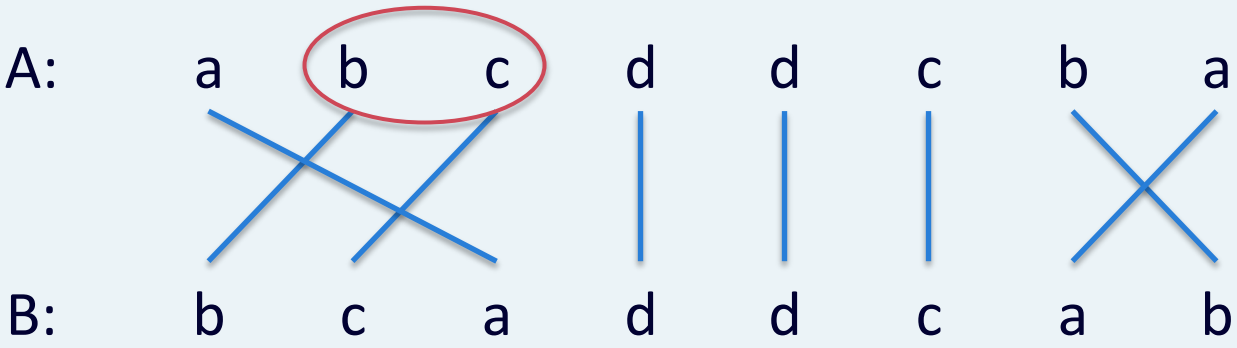
Max Duo-preservation String Mapping Problem (MPSM)

- Input: two strings which are permutations of each other
- Goal: a one-to-one mapping that preserves the maximum number of duos



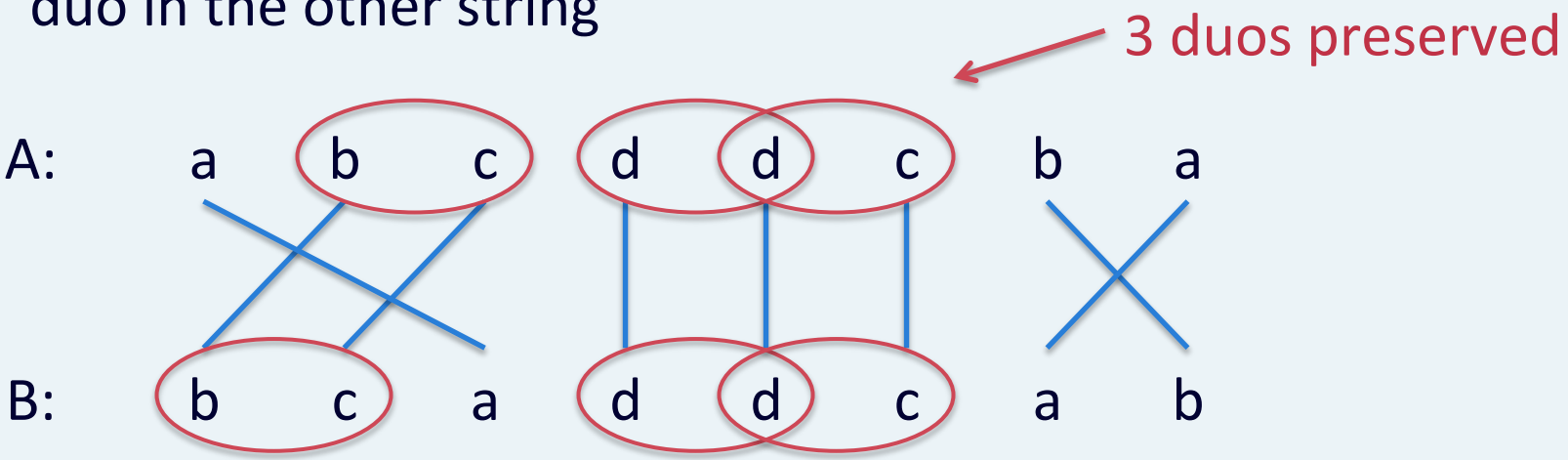
Max Duo-preservation String Mapping Problem (MPSM)

- Input: two strings which are permutations of each other
- Goal: a one-to-one mapping that preserves the maximum number of duos
- A duo is a pair of consecutive letters in a string



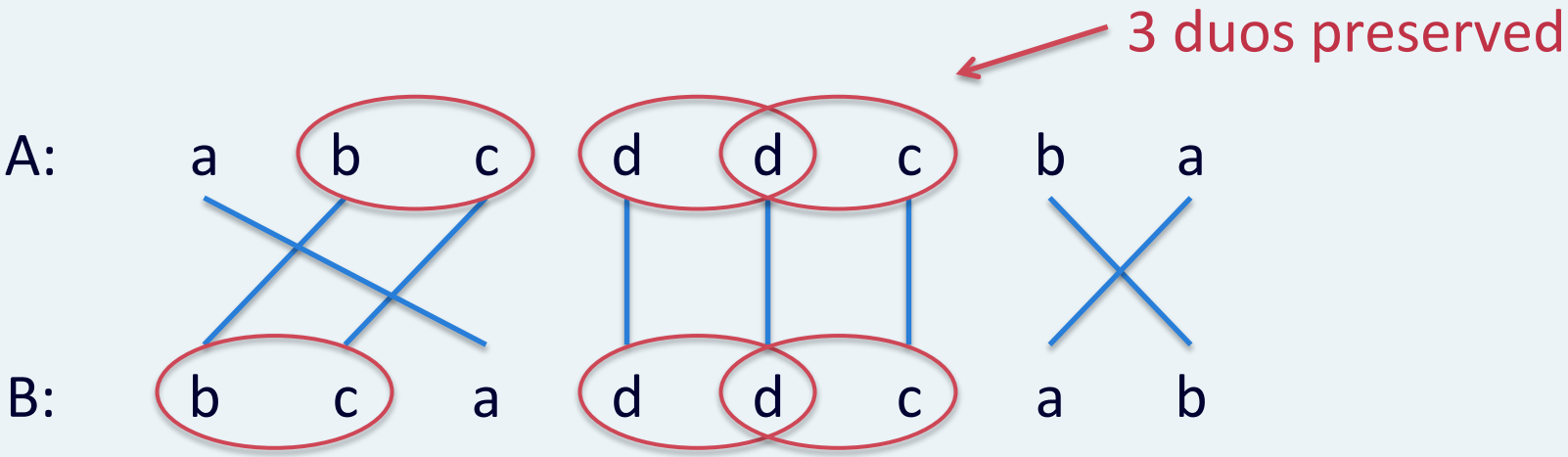
Max Duo-preservation String Mapping Problem (MPSM)

- Input: two strings which are permutations of each other
- Goal: a one-to-one mapping that preserves the maximum number of duos
- A duo is a pair of consecutive letters in a string
- A duo is preserved if it's letters are mapped to some identical duo in the other string



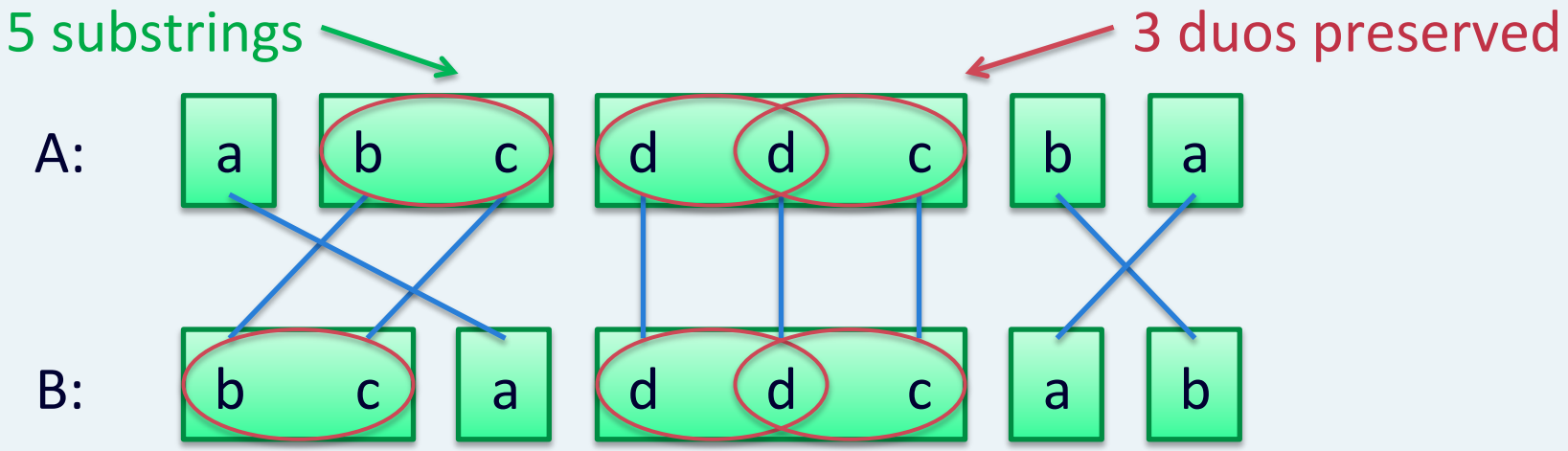
Max Duo-preservation String Mapping Problem (MPSM)

- Complement of the well-studied Minimum Common String Partition problem (MCSP)



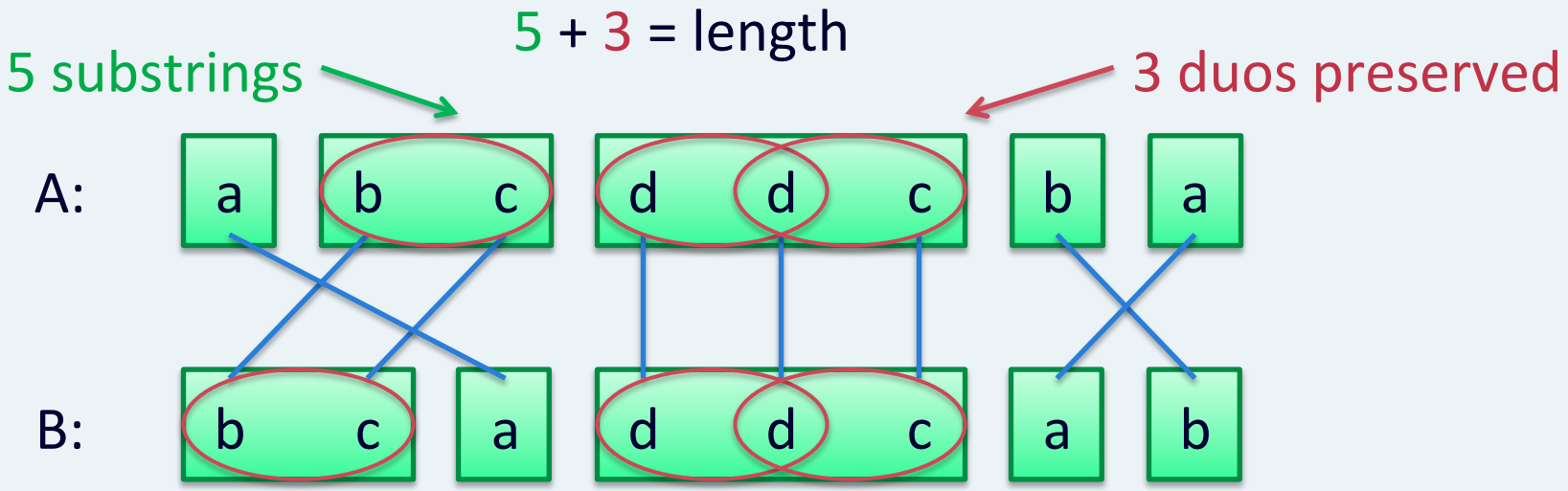
Max Duo-preservation String Mapping Problem (MPSM)

- Complement of the well-studied Minimum Common String Partition problem (MCSP)
  - Separate the strings into identical partitions of min cardinality



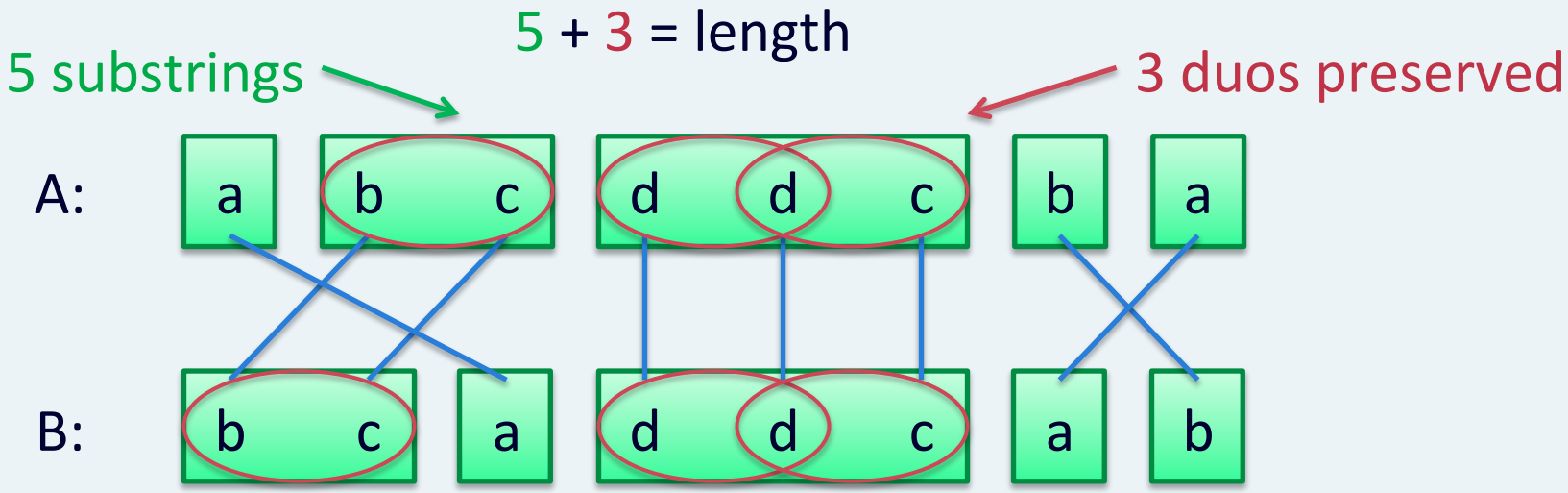
Max Duo-preservation String Mapping Problem (MPSM)

- Complement of the well-studied Minimum Common String Partition problem (MCSP)
  - Separate the strings into identical partitions of min cardinality



Max Duo-preservation String Mapping Problem (MPSM)

- Complement of the well-studied Minimum Common String Partition problem (MCSP)
  - Separate the strings into identical partitions of min cardinality
- In the *k*-MPSM and *k*-MCSP problems, each letter occurs at most *k* times in each string



Related Work

Problem	Approx.	Authors/year
k-MPSM	k^2	Chen, Chen, Samatova, Peng Wang, and Tang (2014)
MPSM	4	Boria, Kurpisz, Leppänen, and Mastrolilli (2014)
	3.5	Boria, Cabodi, Camurati, Palena, Pasini, and Quer (2016)
	3.25	This paper (2016)
MCSP	O(log n log*n)	Cormode and Muthukrishnan (2002) (edit distance with moves problem)

Hardness

- 2-MCSP is NP-hard and APX-hard
  - Goldstein, Kolman, and Zheng (2004)
  - Implies Duo-preservation is also NP-hard
- 2-MPSM is APX-hard
  - Boria, Kurpisz, Leppänen, and Mastrolilli (2014)

Other Related Work

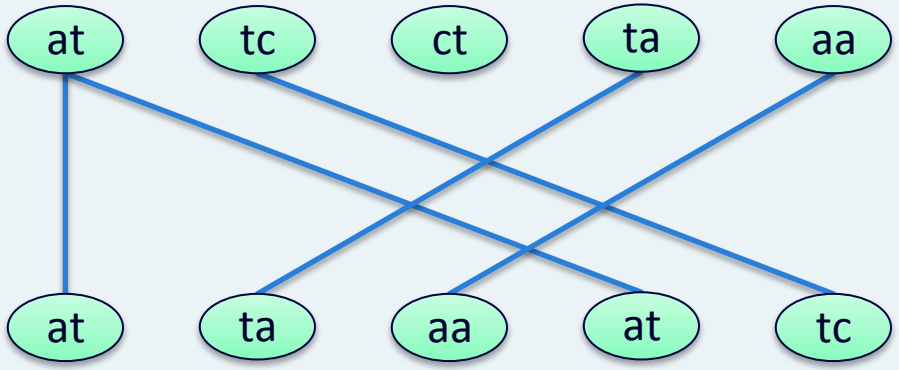
- Fixed parameter tractability (FPT)
  - Duo-preservation is FPT when parameterized by number of preserved duos
    - Beretta, Castelli, Dondi (2015)
- Heuristics (for String Partition)
  - Blum, Lozano, Davidson (2015)
  - Ferdous, Rahman (2015)
- Applications (for String Partition)
  - Chen, Zheng, Fu, Nan, Zhong, Lonardi, Jiang (2005)
  - Swenson, Marron, Earnest-deyoung, Moret (2005)

Duo Mapping and Conflicts

- Consider a bipartite graph on duos
  - Add edges between identical duos
  - Max matching is upper bound on max duo-preservation

A: a t c t a a

B: a t a a t c

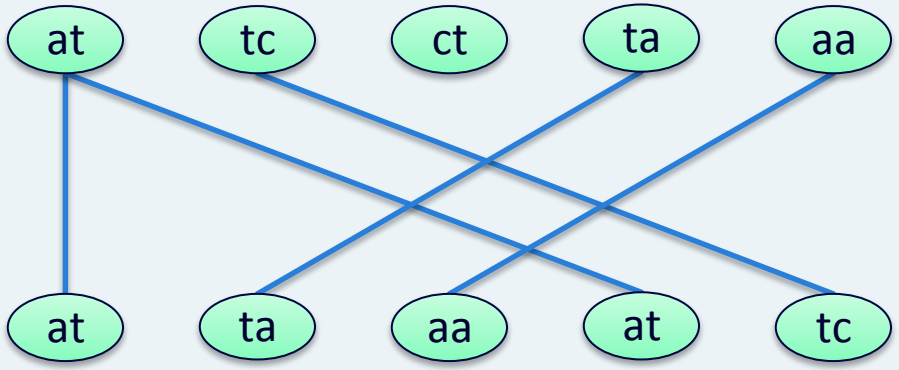


Duo Mapping and Conflicts

- Consider a bipartite graph on duos
  - Add edges between identical duos
  - Max matching is upper bound on max duo-preservation
- Two types of conflicts:

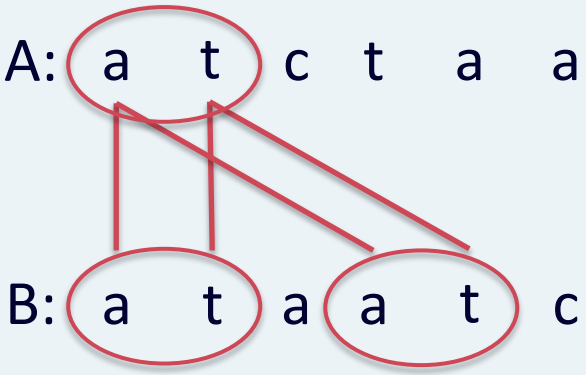
A: a t c t a a

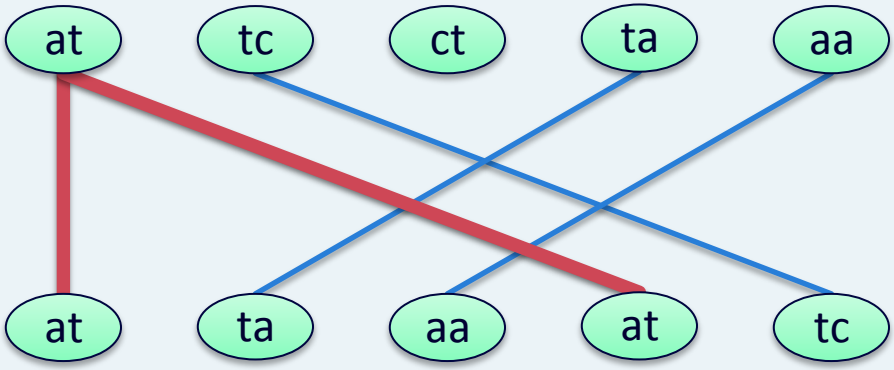
B: a t a a t c



Duo Mapping and Conflicts

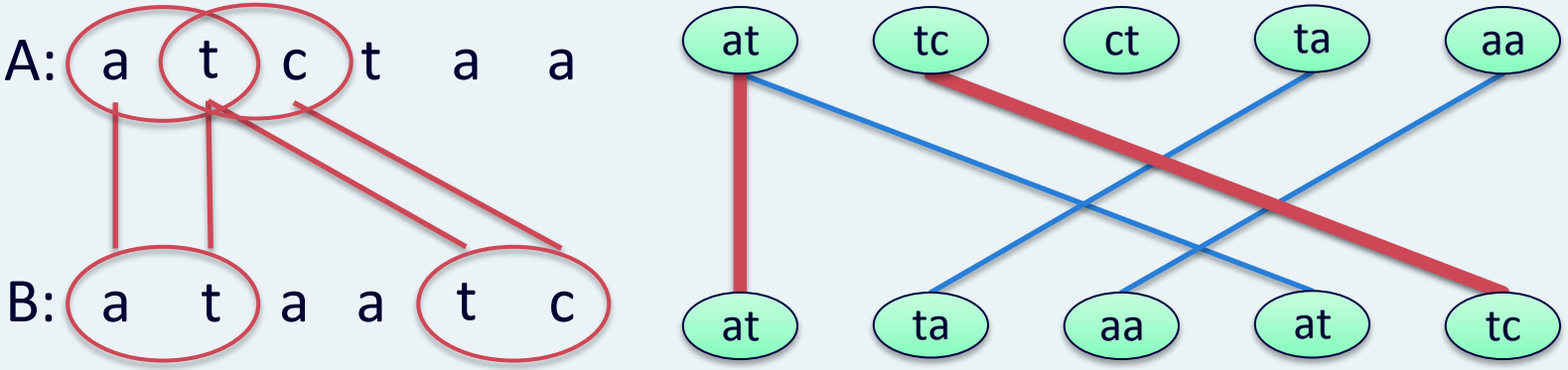
- Consider a bipartite graph on duos
  - Add edges between identical duos
  - Max matching is upper bound on max duo-preservation
- Two types of conflicts:
  - Type 1: two edges to the same duo





Duo Mapping and Conflicts

- Consider a bipartite graph on duos
  - Add edges between identical duos
  - Max matching is upper bound on max duo-preservation
- Two types of conflicts:
  - Type 1: two edges to the same duo
  - Type 2: from consecutive duos to non-consecutive duos



Main Contributions

- New way to construct a duo matching
  - First find weighted matching on triplets
  - Translate to a fractional duo matching
  - Gives a tighter upper bound
- New method for resolving conflicts
  - Based on local graph structures

Triplet Algorithm Overview

- **Step 1:**
  - Append special character '&' to strings (& ≠ &)
  - Construct weighted bipartite graph on triplets
- **Step 2:**
  - Find a weighted matching on triplets that upper bounds the max preserved duos
- **Step 3:**
  - Use triplet matching to construct a fractional duo matching
- **Step 4:**
  - Translate edges into a string mapping that preserves at least 1/3.25 of the max preserved duos

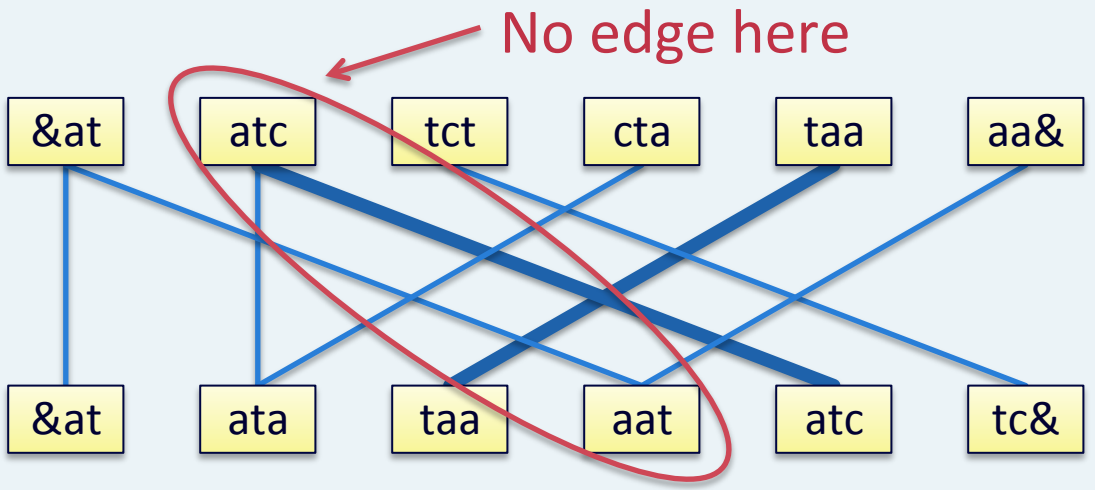
Step 1: Triplet Matching Graph

- If all three characters match, add an edge with weight 1
- If the first two letters of both or last two letters of both match, add an edge with weight 0.5

A: & a t c t a a &
B: & a t a a t c &

= 1

= 0.5



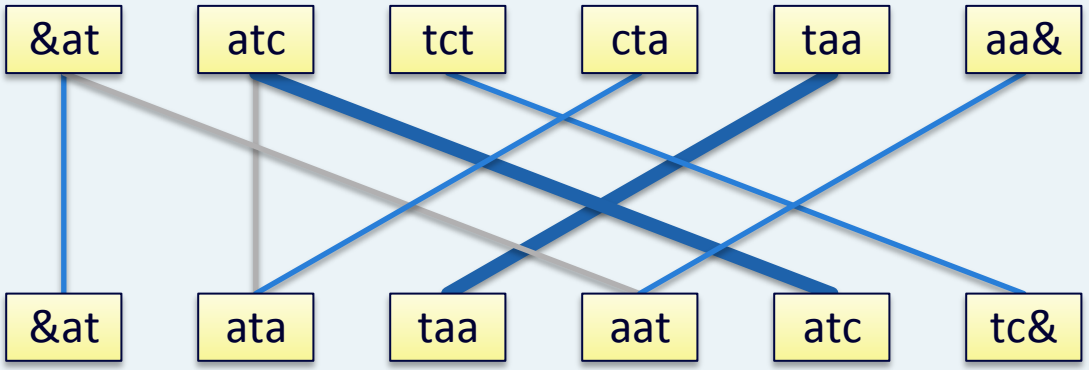
Step 2: Max Weight Triplet Matching

- Find a maximum weight matching
- The weight of this matching upper bounds the maximum number of preserved duos
- Ex: max weight = 0.5 + 1 + 0.5 + 0.5 +1 + 0.5 = 4

A: & a t c t a a &
B: & a t a a t c &

= 1

= 0.5

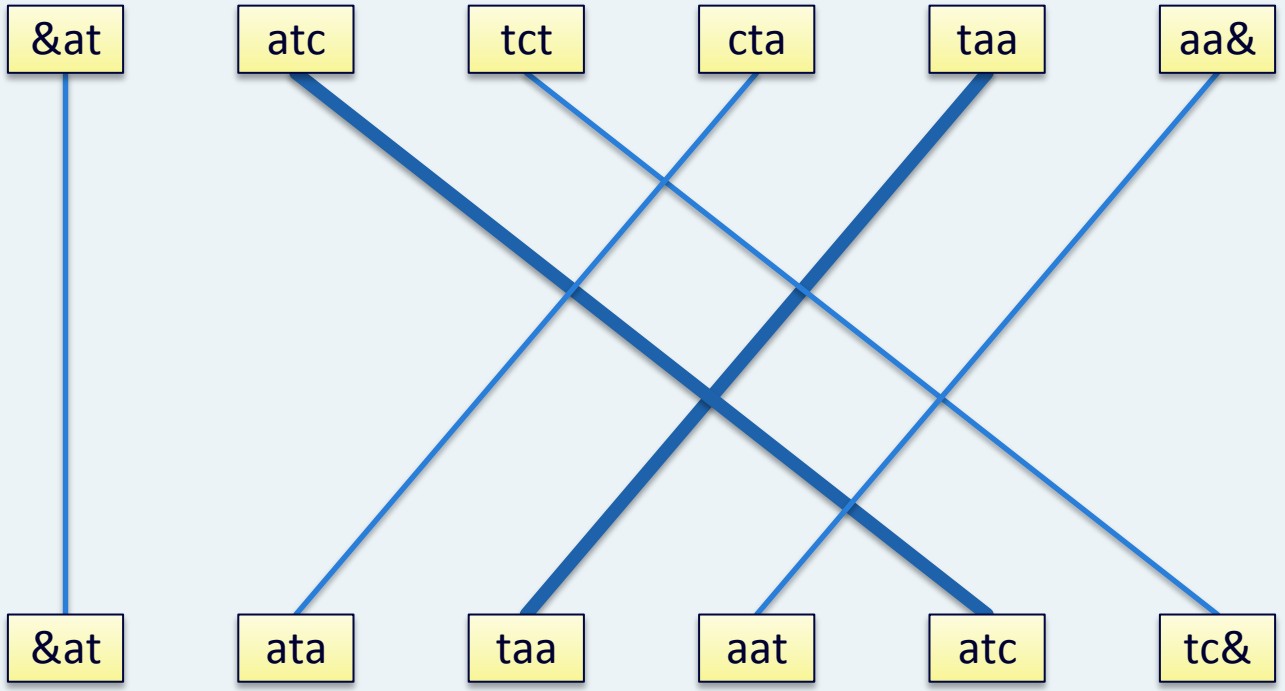


Step 3: Translate to Duo Matching

- Triplet edge weight 1
  - Add 0.5 edge for both duo pairs
- Triplet edge weight 0.5
  - Add 0.5 edge only for duo pair that matches
- If a matching pair of duos gets two weight 0.5 edges, combine into one weight 1 edge

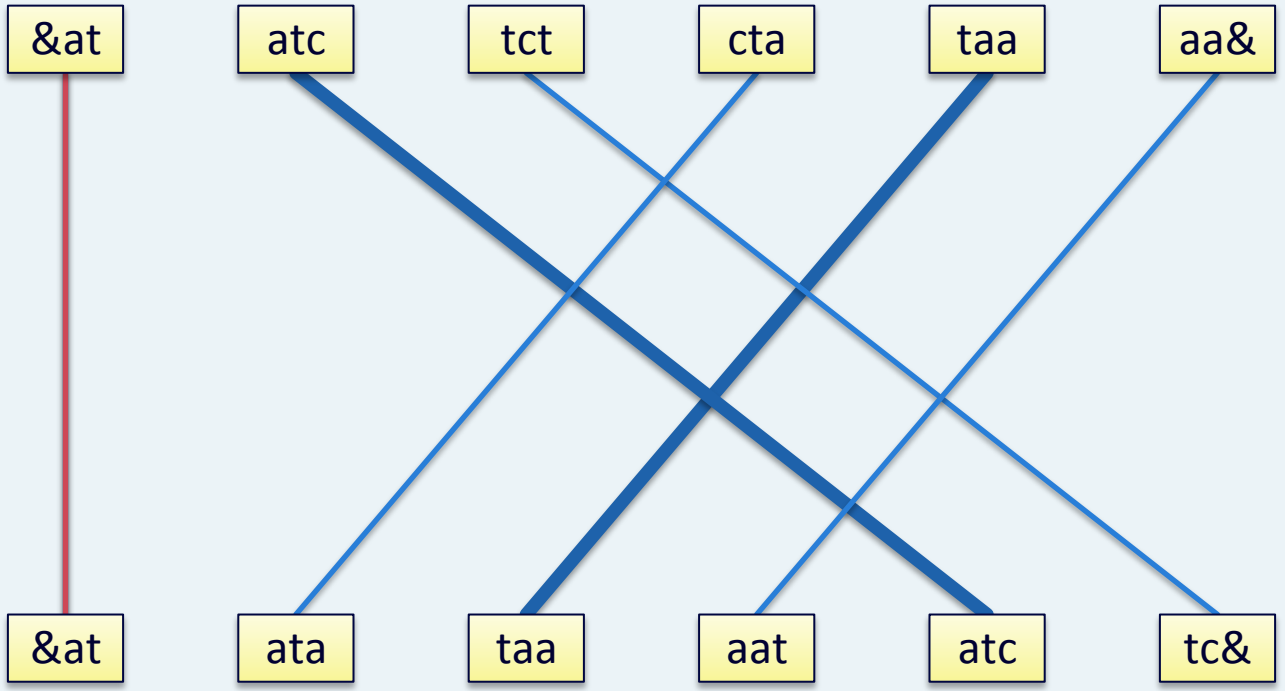
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



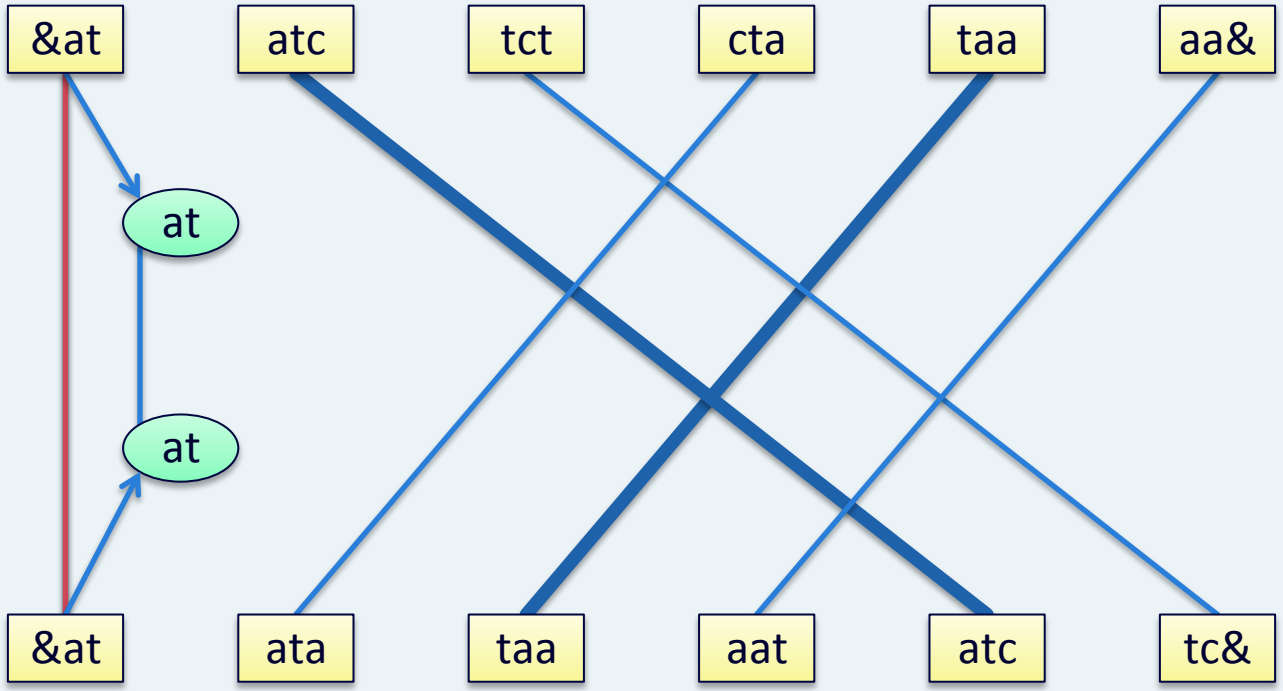
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



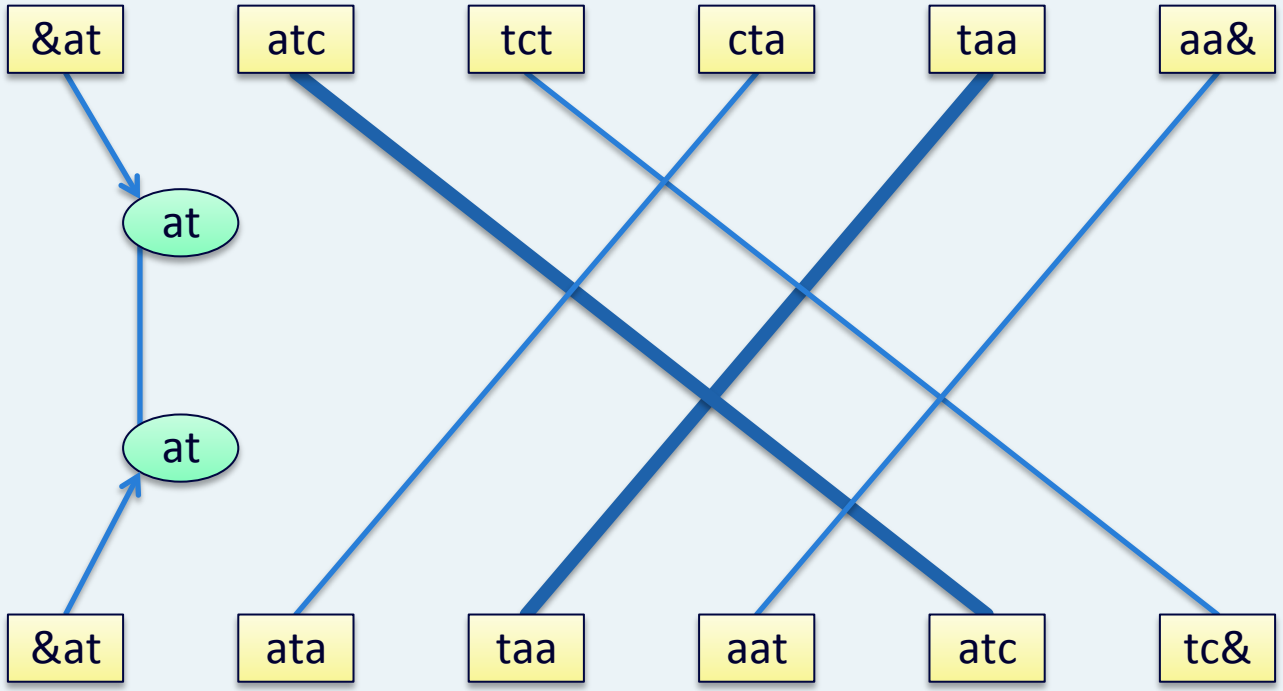
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



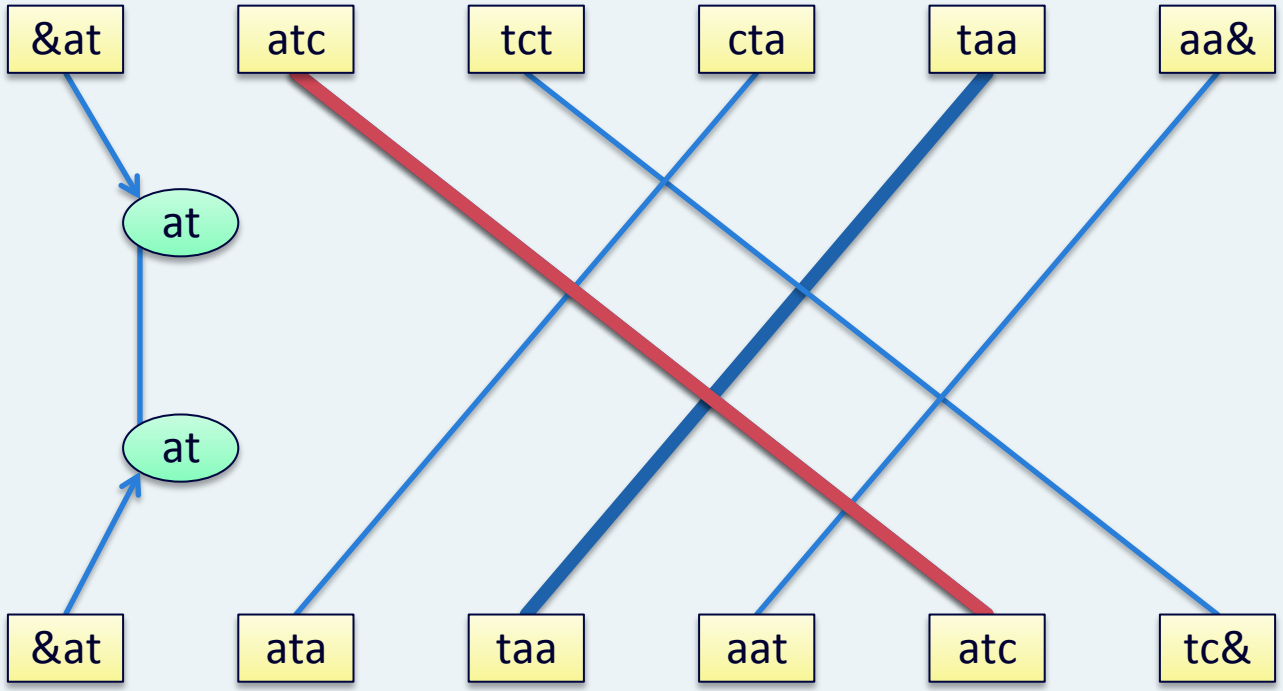
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



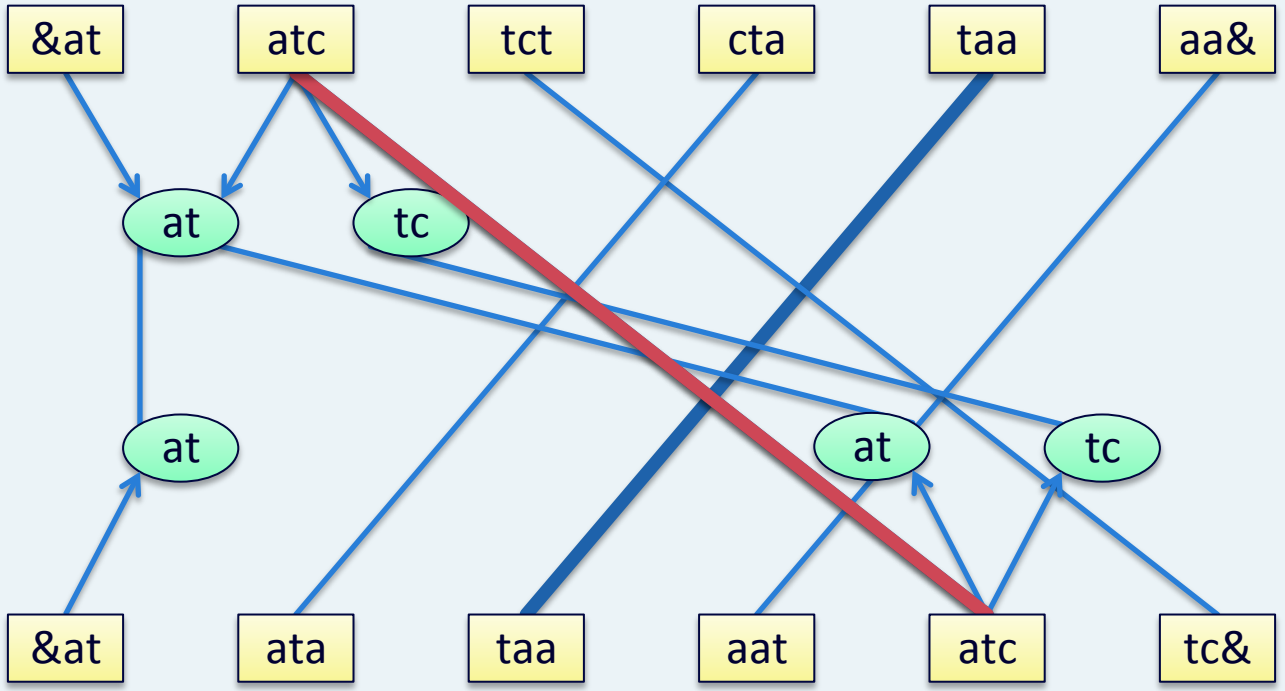
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



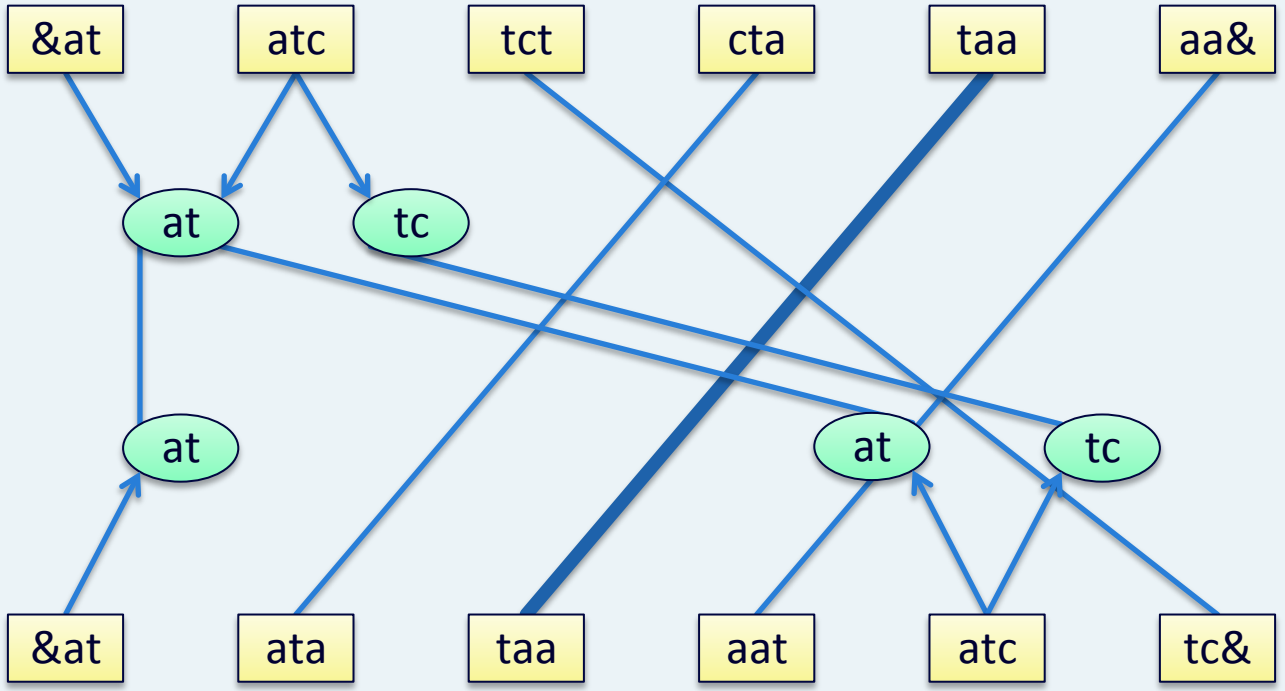
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



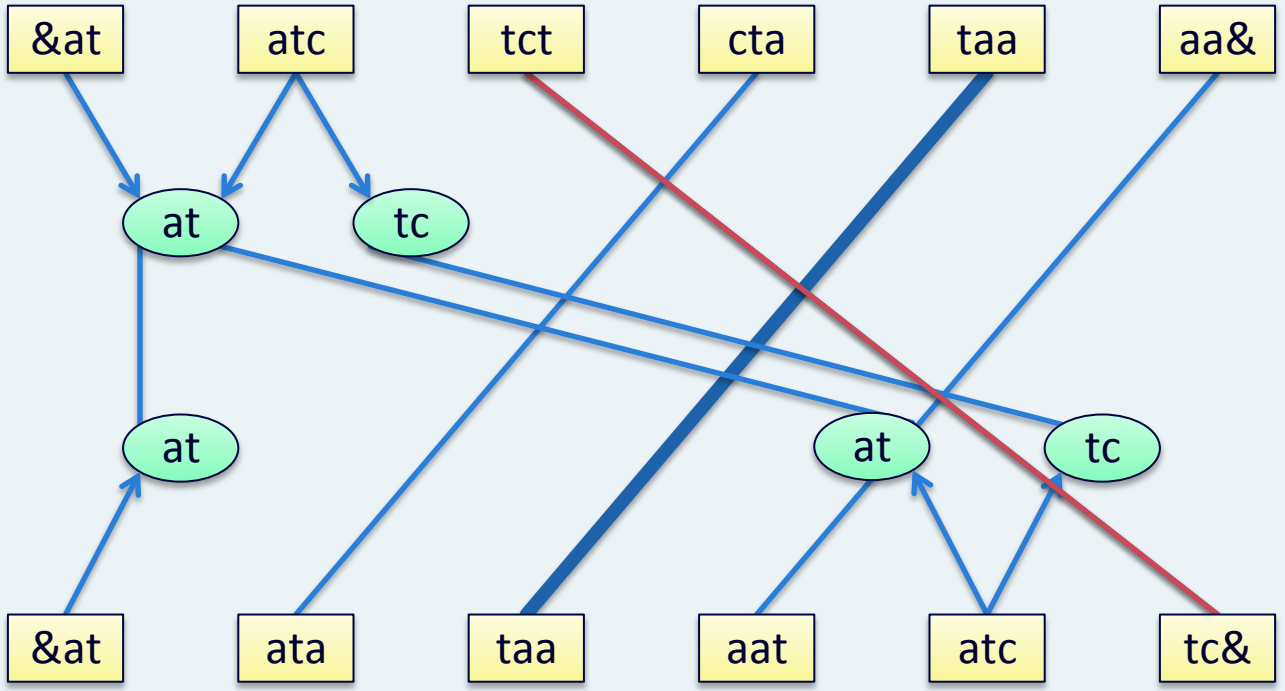
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



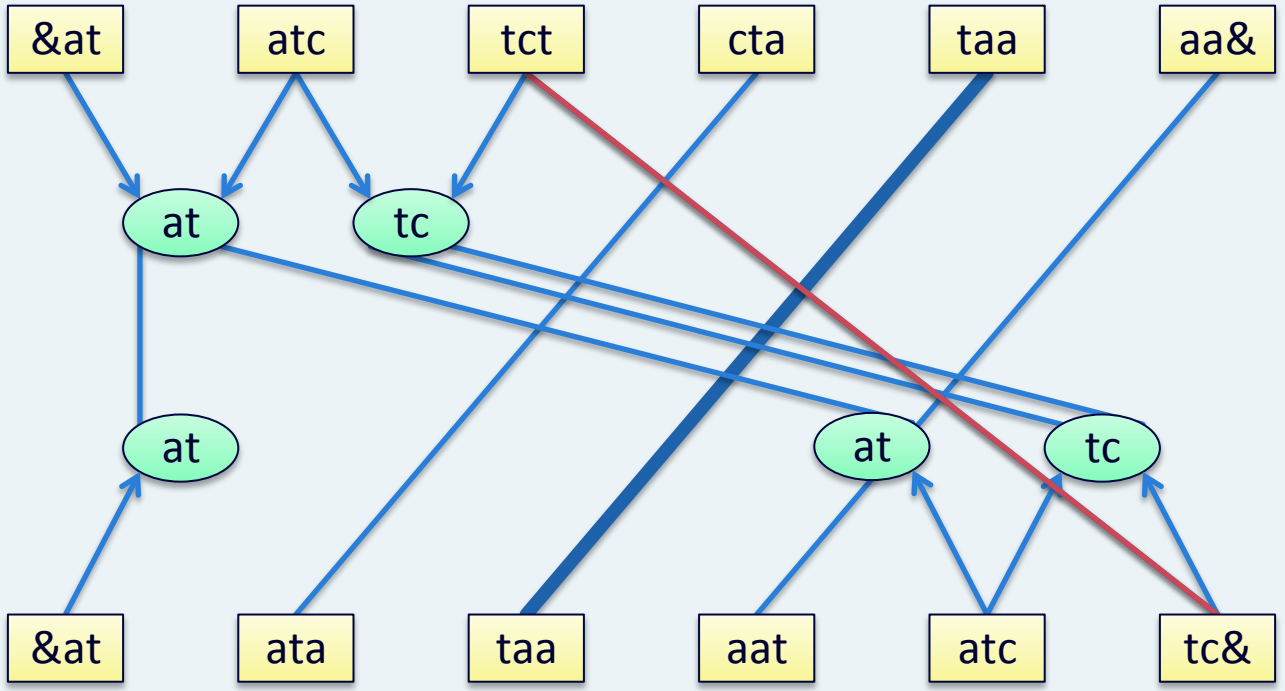
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



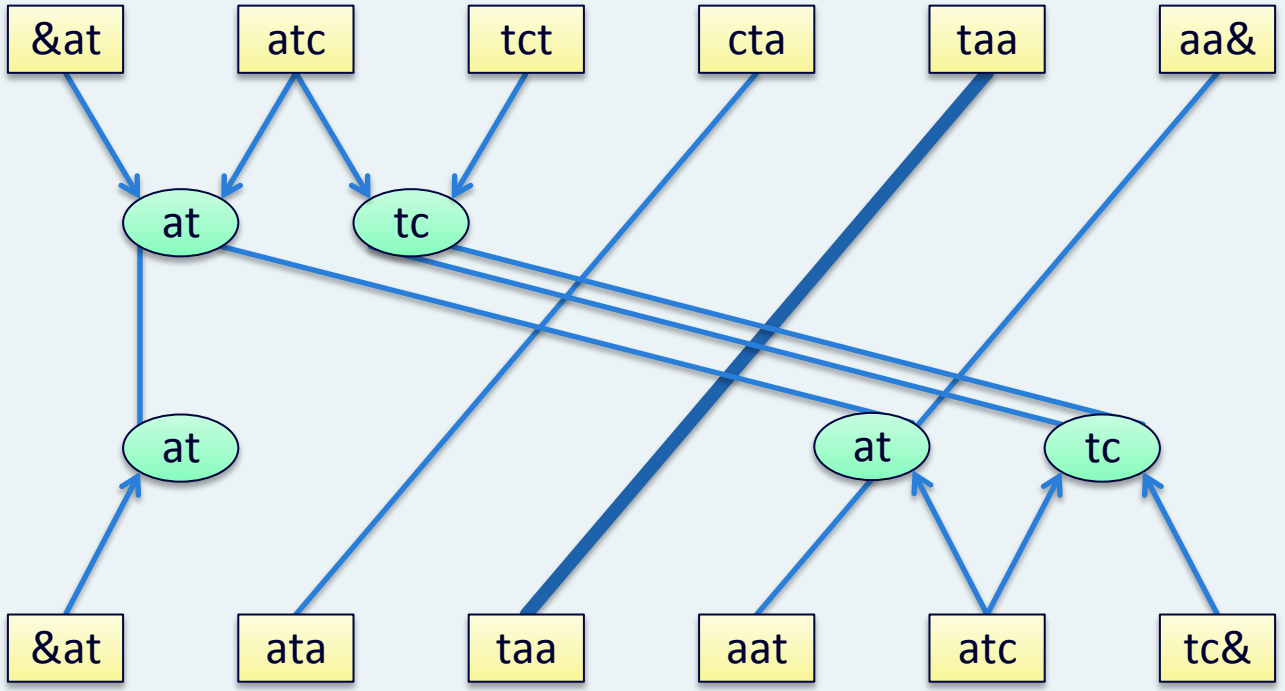
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



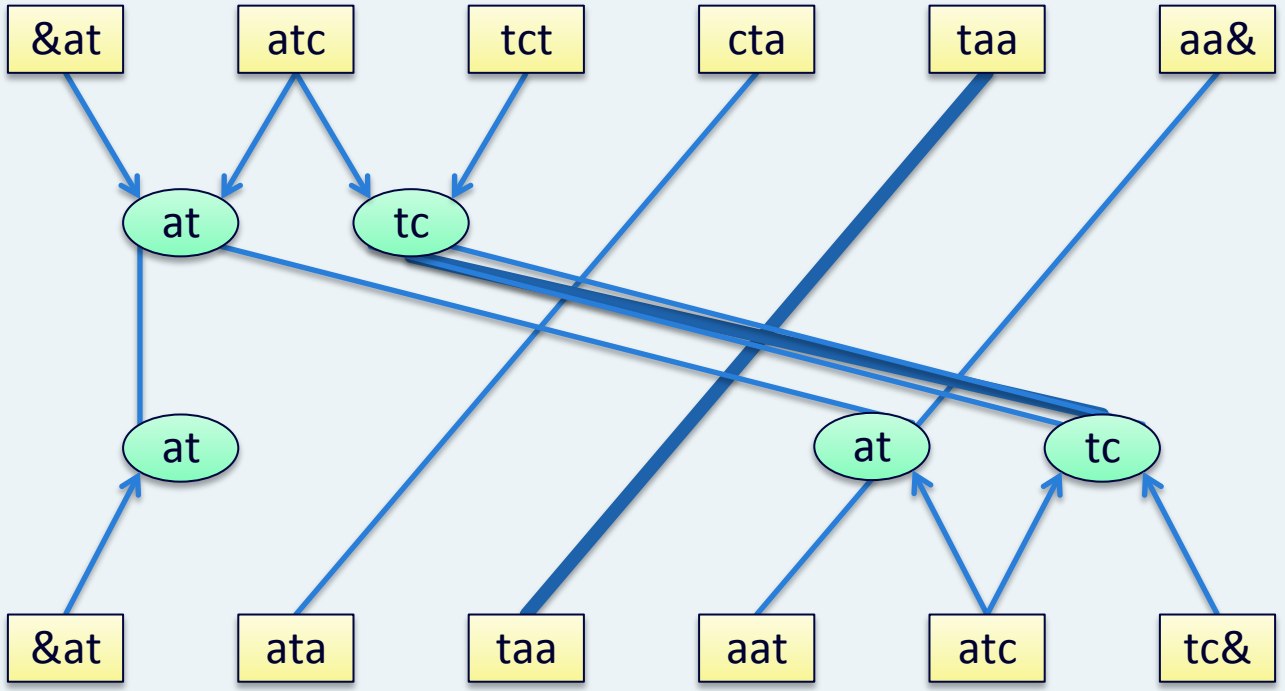
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



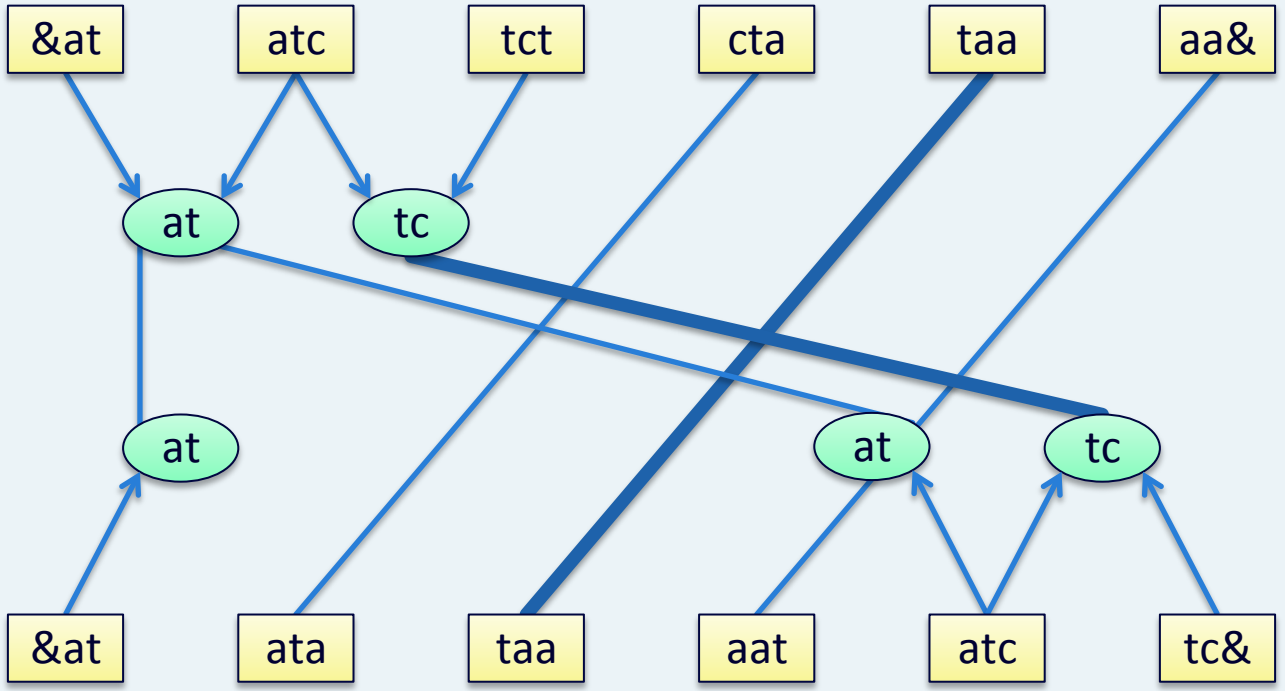
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



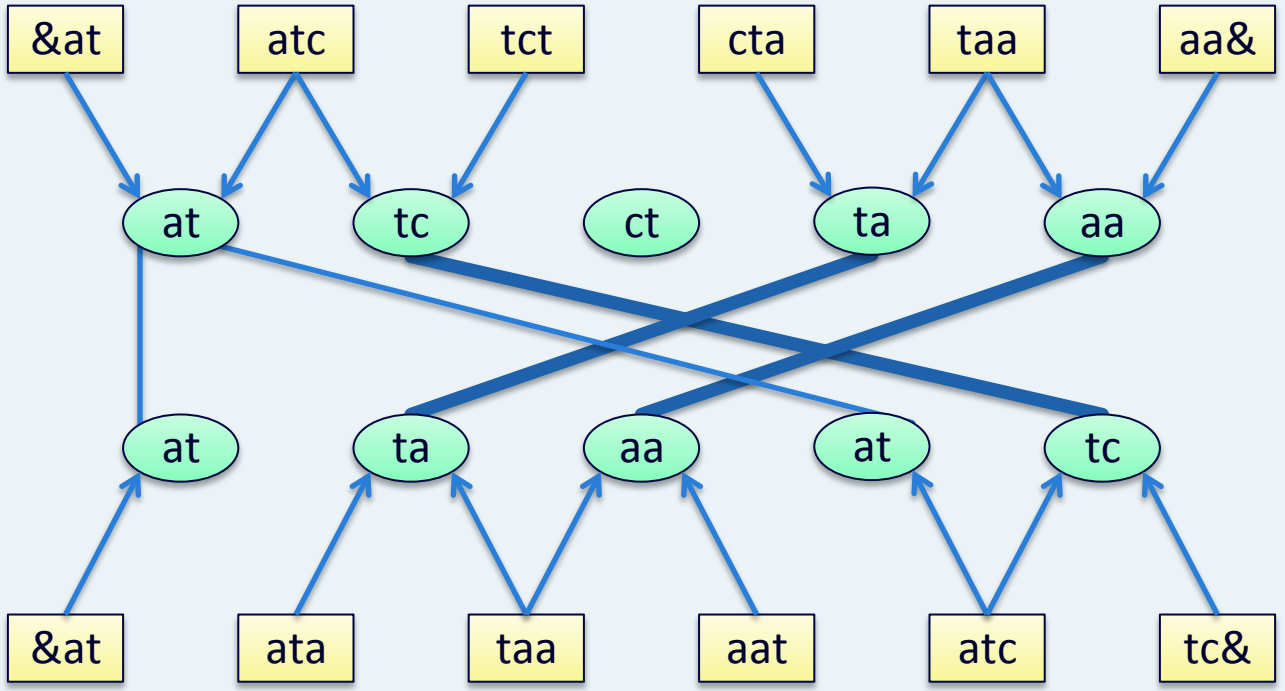
Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



Step 3: Translate to Duo Matching

A: & a t c t a a &
B: & a t a a t c &



Triplet Algorithm Overview

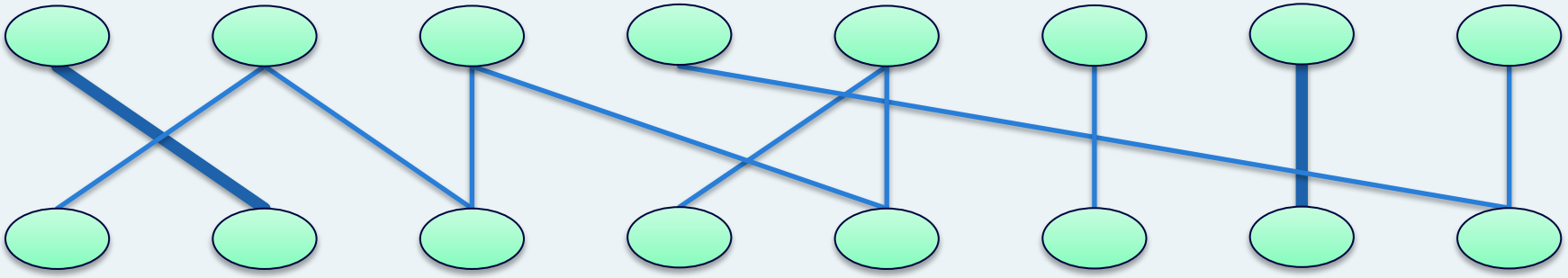
- **Step 1:**
  - Append special character '&' to strings (& ≠ &)
  - Construct weighted bipartite graph on triplets
- **Step 2:**
  - Find a weighted matching on triplets that upper bounds the max preserved duos
- **Step 3:**
  - Use triplet matching to construct a fractional duo matching
- **Step 4:**
  - Translate edges into a string mapping that preserves at least 1/3.25 of the max preserved duos

Step 4: Selecting Edges to Map

- Edges are selected in three phases
  - Selected edge = pair of duos that get mapped to each other
  - Must remove selected edge and conflicting edges from the duo matching
- Each phase considers edges in the order they appear in the first string

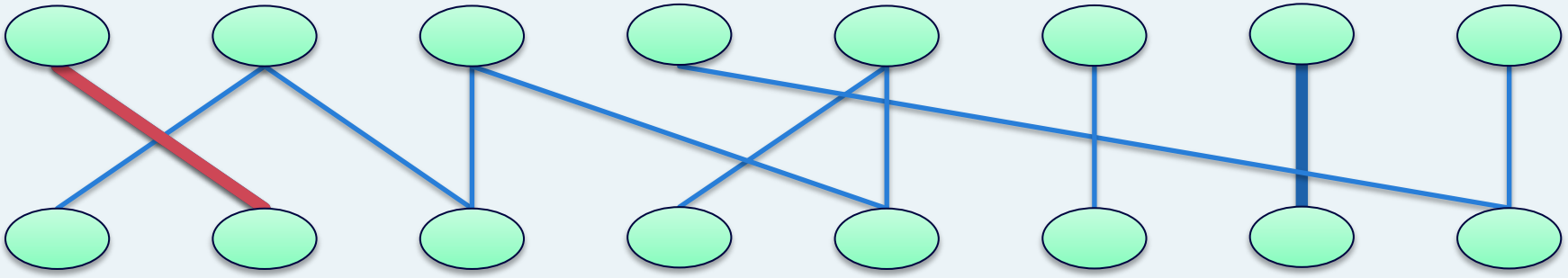
Step 4: Selecting Edges to Map

- Phase 1: Select weight 1 edges
- Phase 2: Select pairs of consecutive parallel edges
- Phase 3: Select any remaining edges



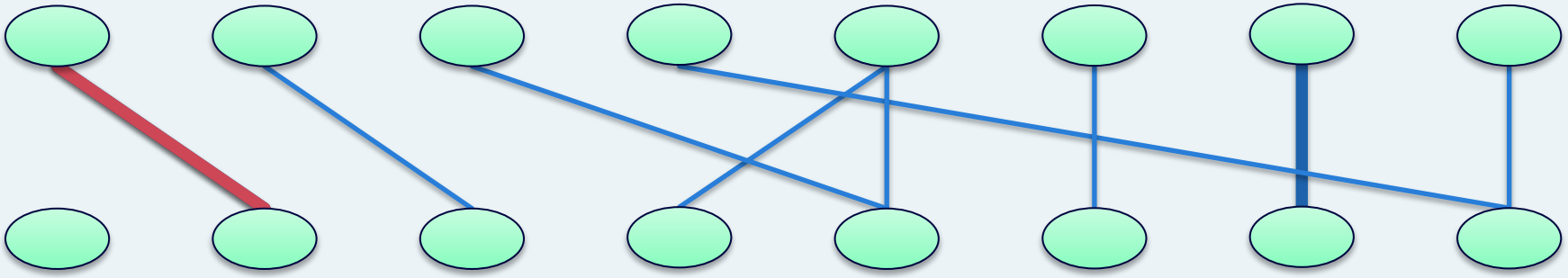
Step 4: Selecting Edges to Map

- Phase 1: Select weight 1 edges
- Phase 2: Select pairs of consecutive parallel edges
- Phase 3: Select any remaining edges



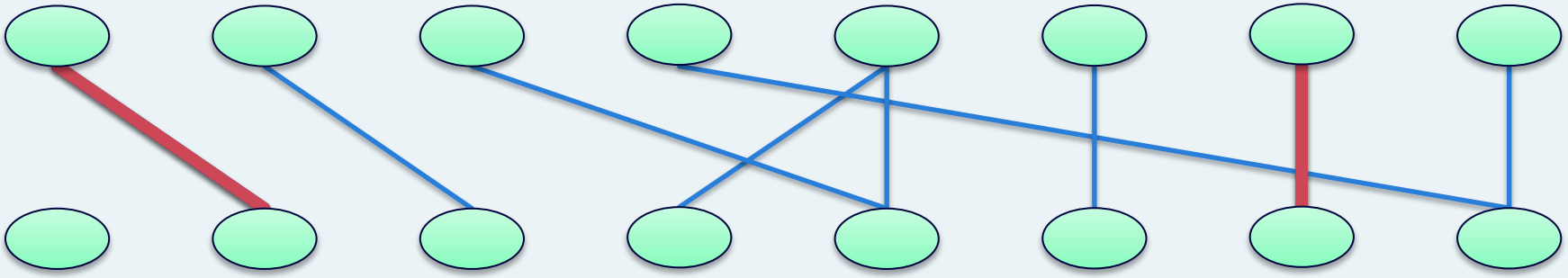
Step 4: Selecting Edges to Map

- Phase 1: Select weight 1 edges
- Phase 2: Select pairs of consecutive parallel edges
- Phase 3: Select any remaining edges



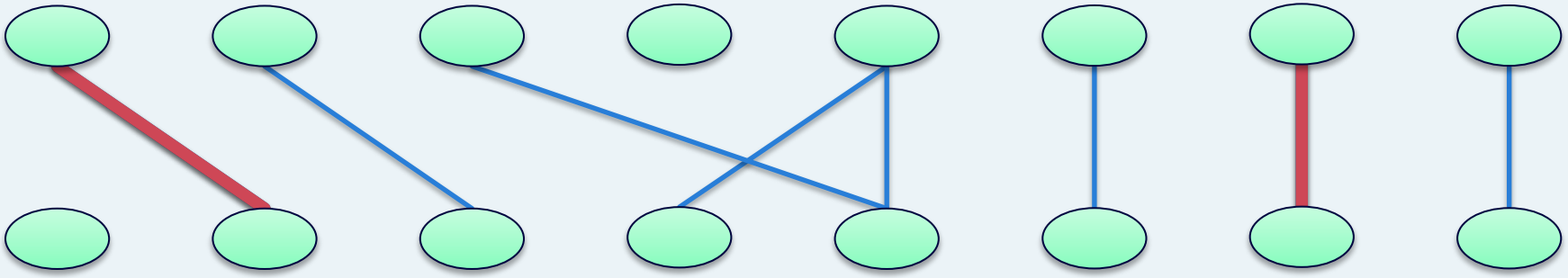
Step 4: Selecting Edges to Map

- Phase 1: Select weight 1 edges
- Phase 2: Select pairs of consecutive parallel edges
- Phase 3: Select any remaining edges



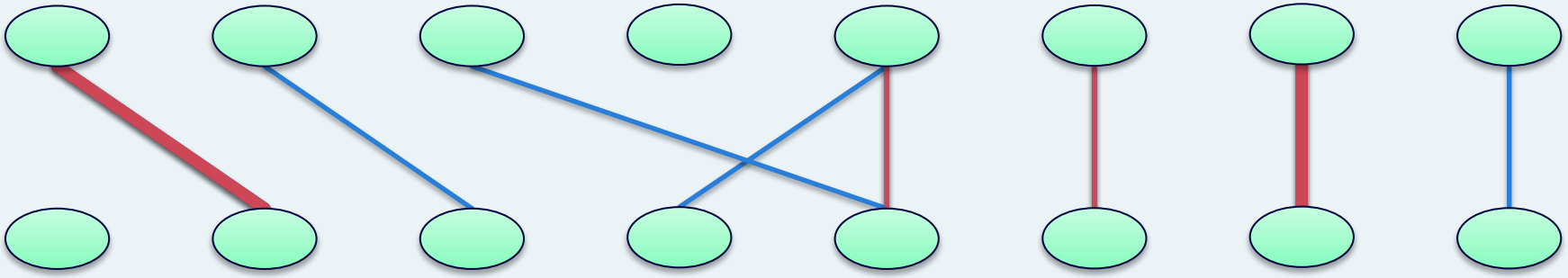
Step 4: Selecting Edges to Map

- Phase 1: Select weight 1 edges
- Phase 2: Select pairs of consecutive parallel edges
- Phase 3: Select any remaining edges



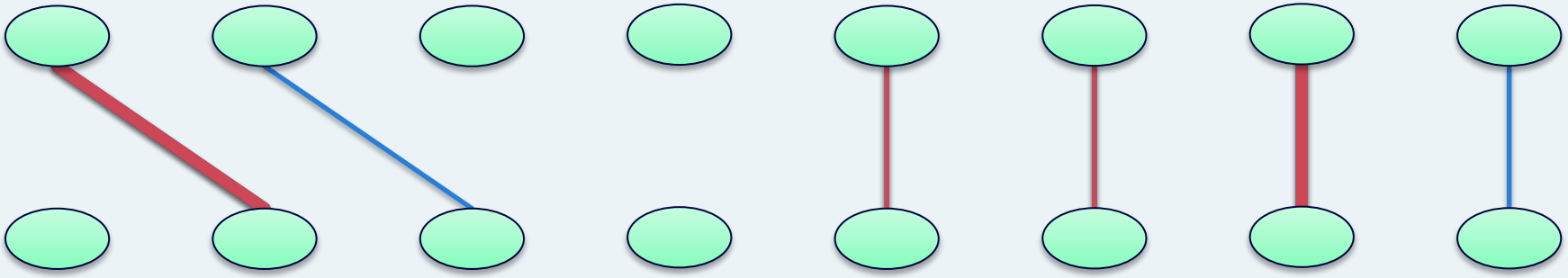
Step 4: Selecting Edges to Map

- Phase 1: Select weight 1 edges
- Phase 2: Select pairs of consecutive parallel edges
- Phase 3: Select any remaining edges



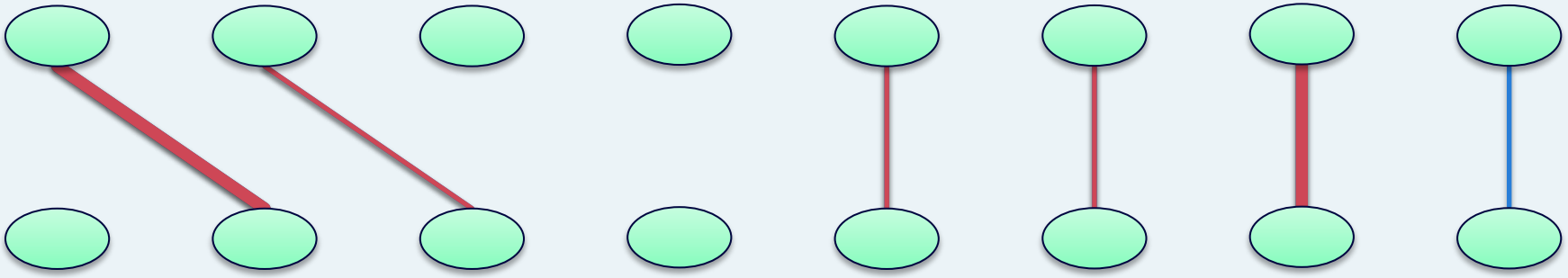
Step 4: Selecting Edges to Map

- Phase 1: Select weight 1 edges
- Phase 2: Select pairs of consecutive parallel edges
- Phase 3: Select any remaining edges



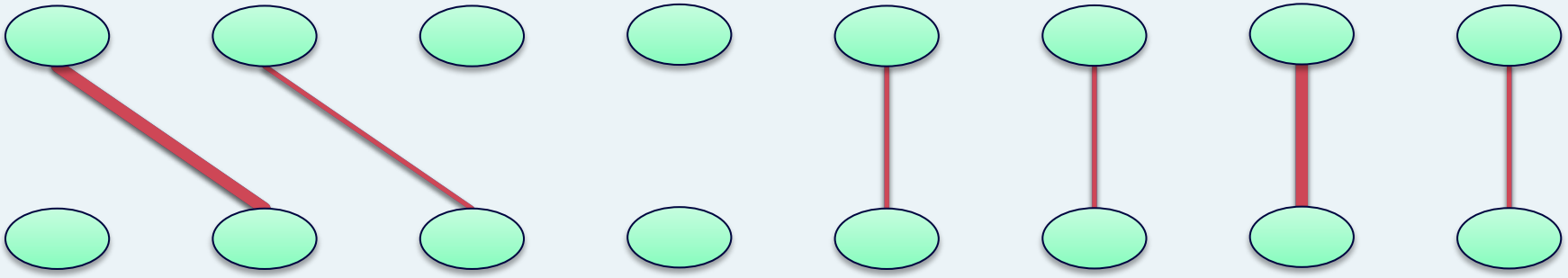
Step 4: Selecting Edges to Map

- Phase 1: Select weight 1 edges
- Phase 2: Select pairs of consecutive parallel edges
- Phase 3: Select any remaining edges

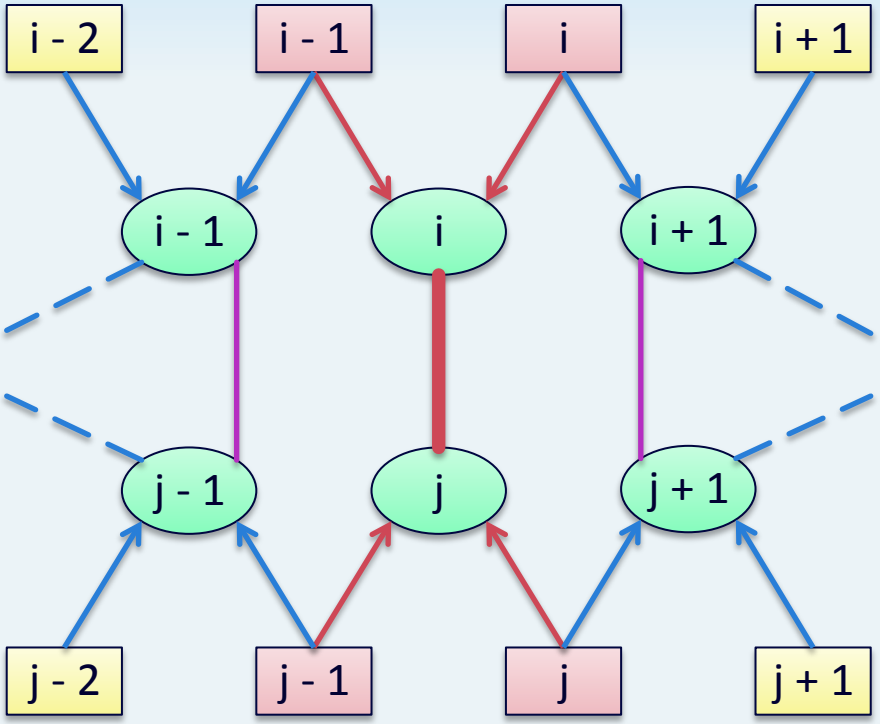


Step 4: Selecting Edges to Map

- Phase 1: Select weight 1 edges
- Phase 2: Select pairs of consecutive parallel edges
- Phase 3: Select any remaining edges

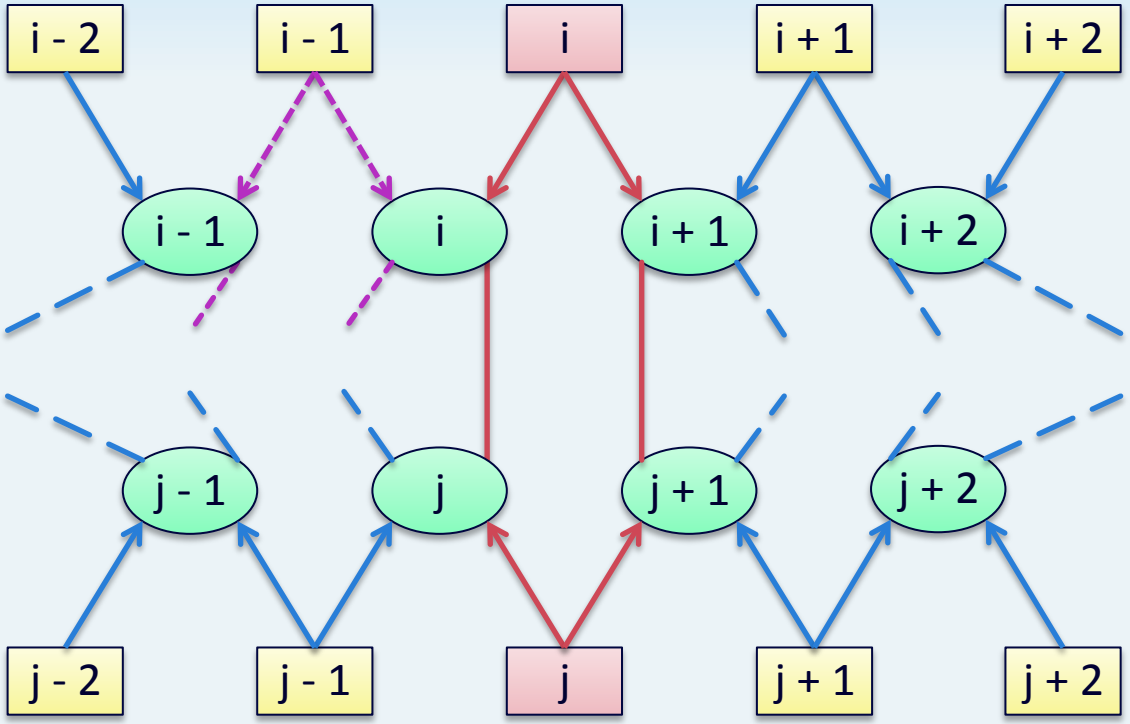


Proof-by-pictures: Step 4, Phase 1



- Solid purple edges can't be conflicting, aren't removed
- Total weight removed = 4(1/2) + 1 = 3
- Ratio = 1/3

Proof-by-pictures: Step 4, Phase 2



- Only one of the two purple edges from i - 1 can exist
- Total weight removed = 10(1/2) + 1(1/2) + 2(1/2) = 6.5
- Ratio = 2/6.5 = 1/3.25

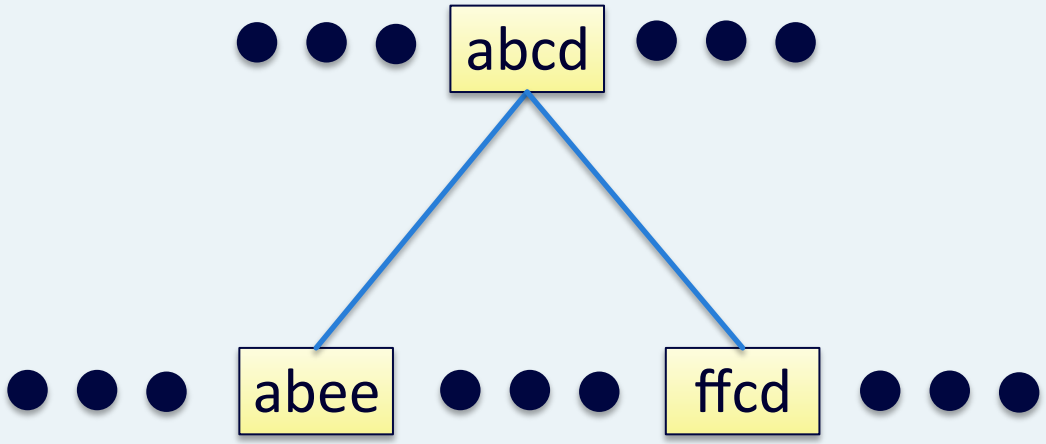
Future Work

- 2 or 3-approximation
  - Can this work be combined with local search techniques of Boria *et al* (2016)?
- Connecting to more applications
- Heuristics

Thank You!

Why not 4-mers?

- Appears that a max weight matching would not upper bound the max preserved duos
  - Can only choose one edge for a 4-mer matching



Why not 4-mers?

- Appears that a max weight matching would not upper bound the max preserved duos
  - Can only choose one edge for a 4-mer matching
  - Duo mapping can match “ab” and “cd” separately