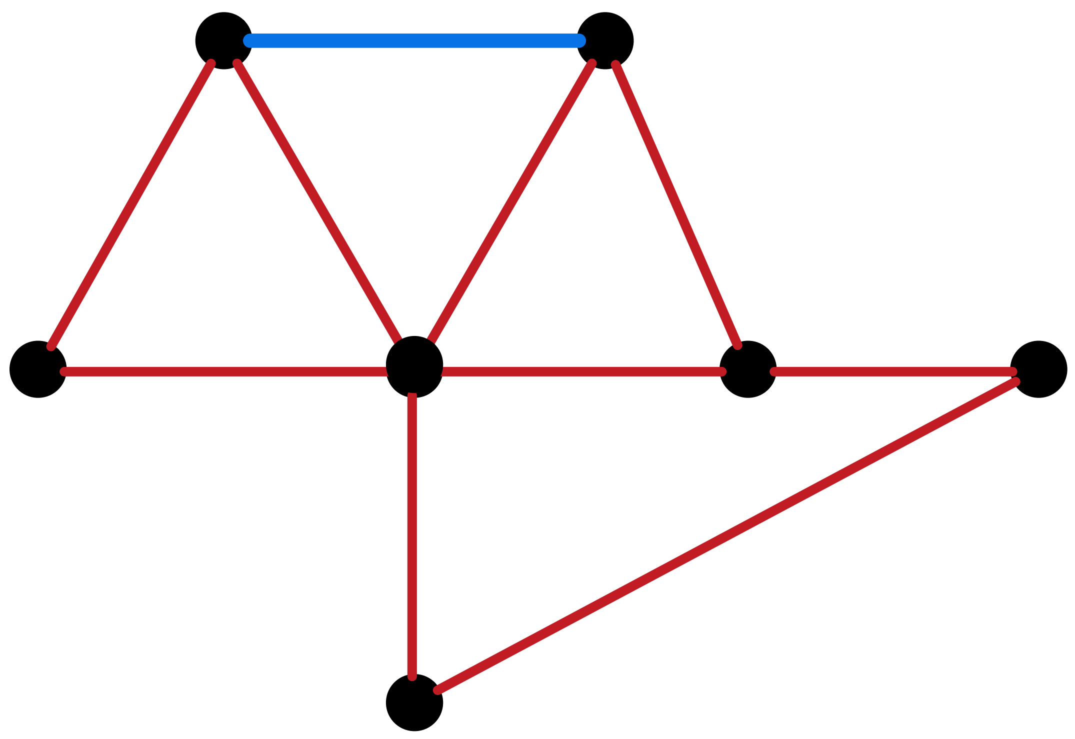
PATH CONTRACTION FASTER THAN 2^n

ICALP 2019, Patras, Greece

Akanksha Agrawal, Fedor Fomin, Daniel Lokshtanov, Saket Saurabh and Prafullkumar Tale

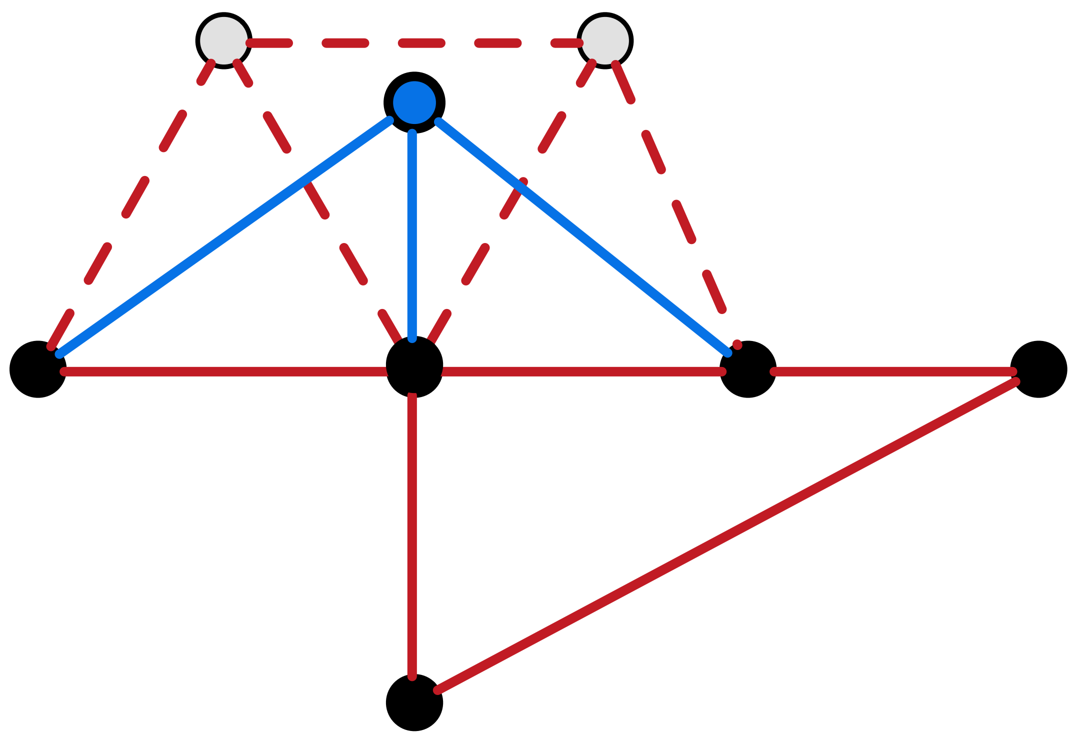
Ben-Gurion University of the Negev

CONTRACTION OF AN EDGE



- Delete the two end-points.

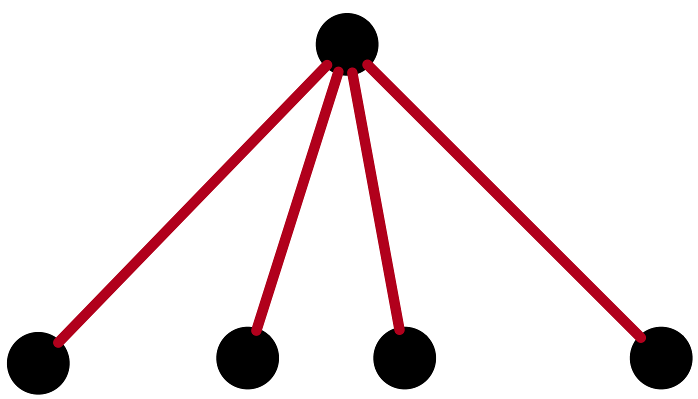
CONTRACTION OF AN EDGE



- Delete the two end-points.
- Add a new vertex adjacent to the neighbors of the deleted vertices.

CONTRACTION TO A PATH

P_t = Path on t vertices



P_3 P_4

Contract a subset E^* of $E(G)$ such that the resulting graph is isomorphic to P_t.

PATH CONTRACTION

Input: Graph G.

Output: Maximum number t, such that G can be contracted to a path on t vertices.

SOME RESULTS ON PATH CONTRACTION

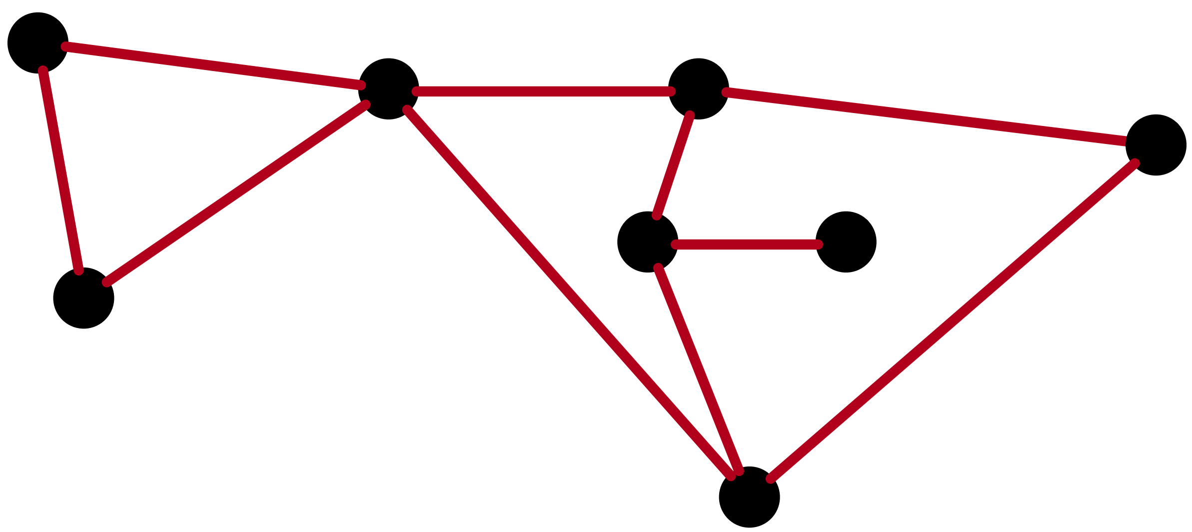
- NP-hard, for every fixed t >= 4.
- Polynomial time solvable for t <= 3.
- The problem has been well studied for restricted input graphs.
- The case when t = 4 is the same as 2-DISJOINT CONNECTED SUBGRAPHS, which admits an algorithm running in time $O^*(1.7804^n)$.

---

t is the length of the contracted path.

INTERPRETING CONTRACTION AS PARTITIONS

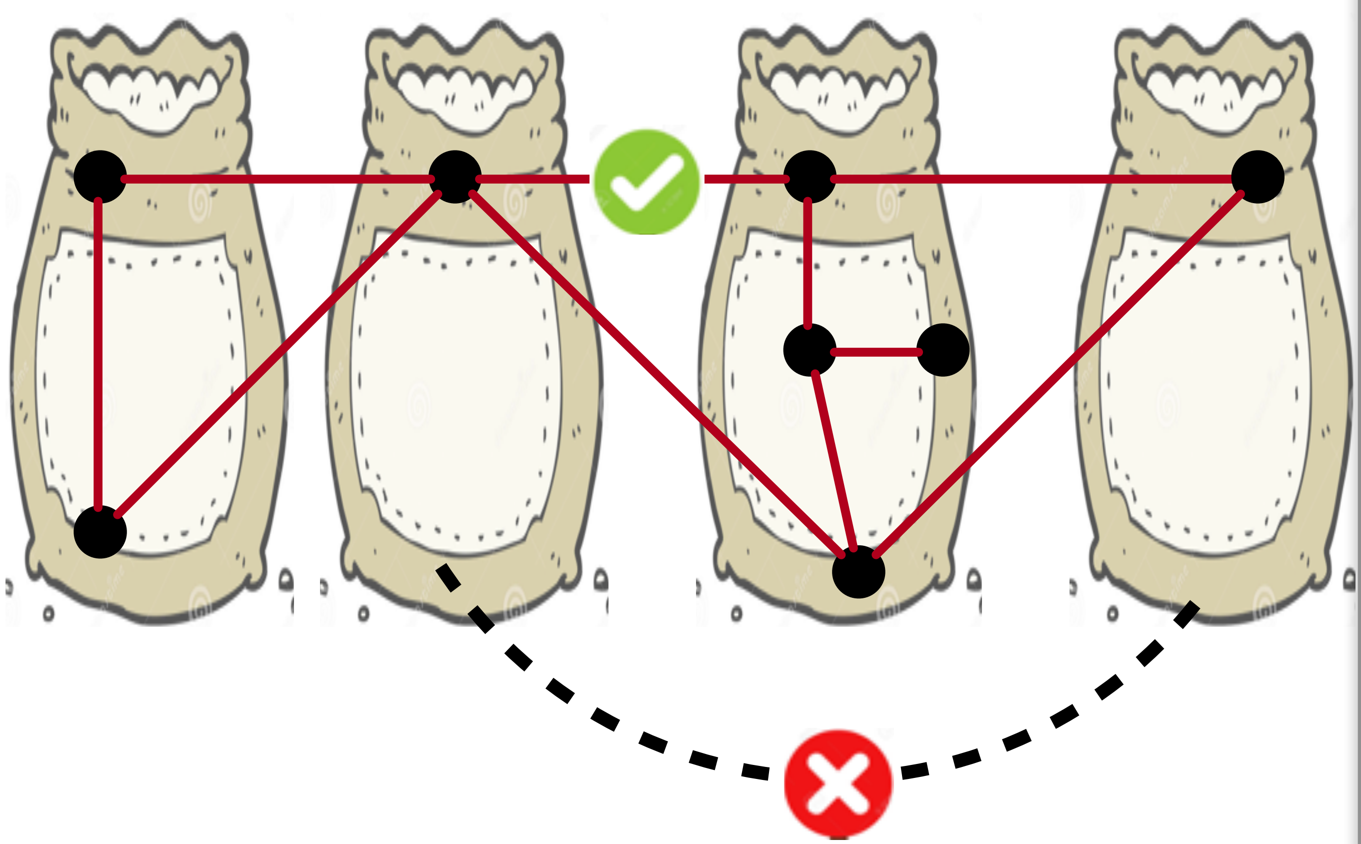
P_t = Path on t vertices



A partition of vertices, where edges exits only between consecutive bags.

INTERPRETING CONTRACTION AS PARTITIONS

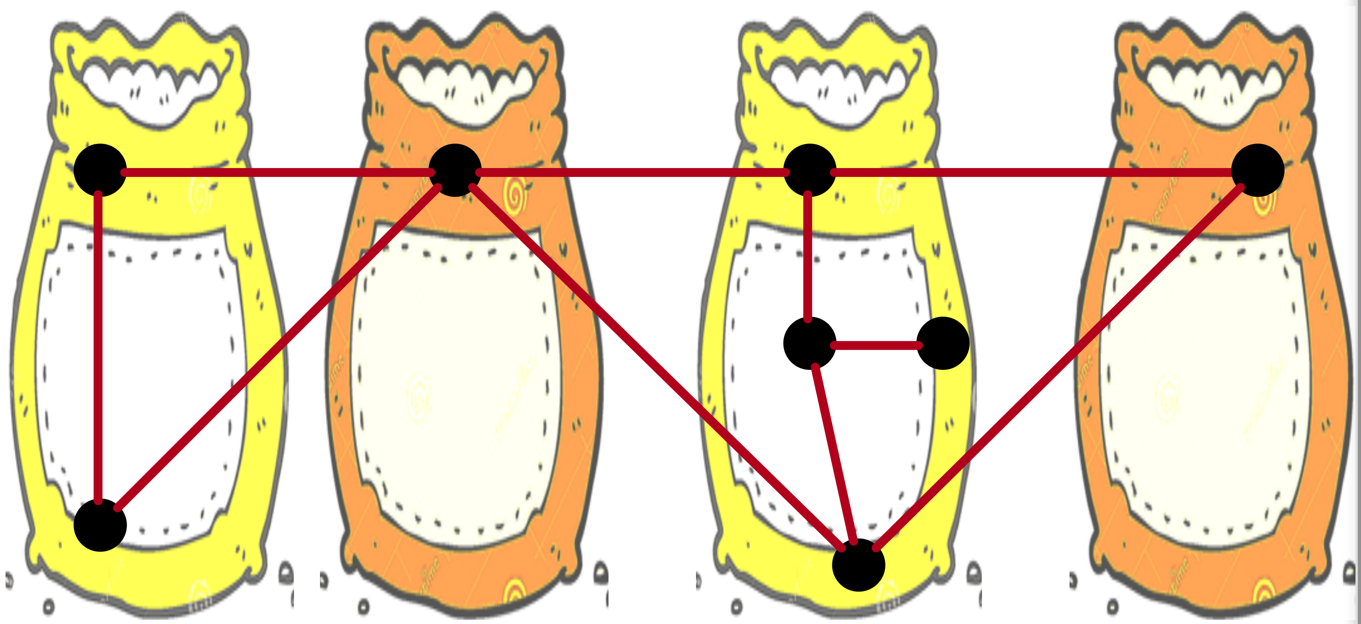
P_t = Path on t vertices



A partition of vertices, where edges exits only between consecutive bags.

STARTING POINT OF THE RESULT

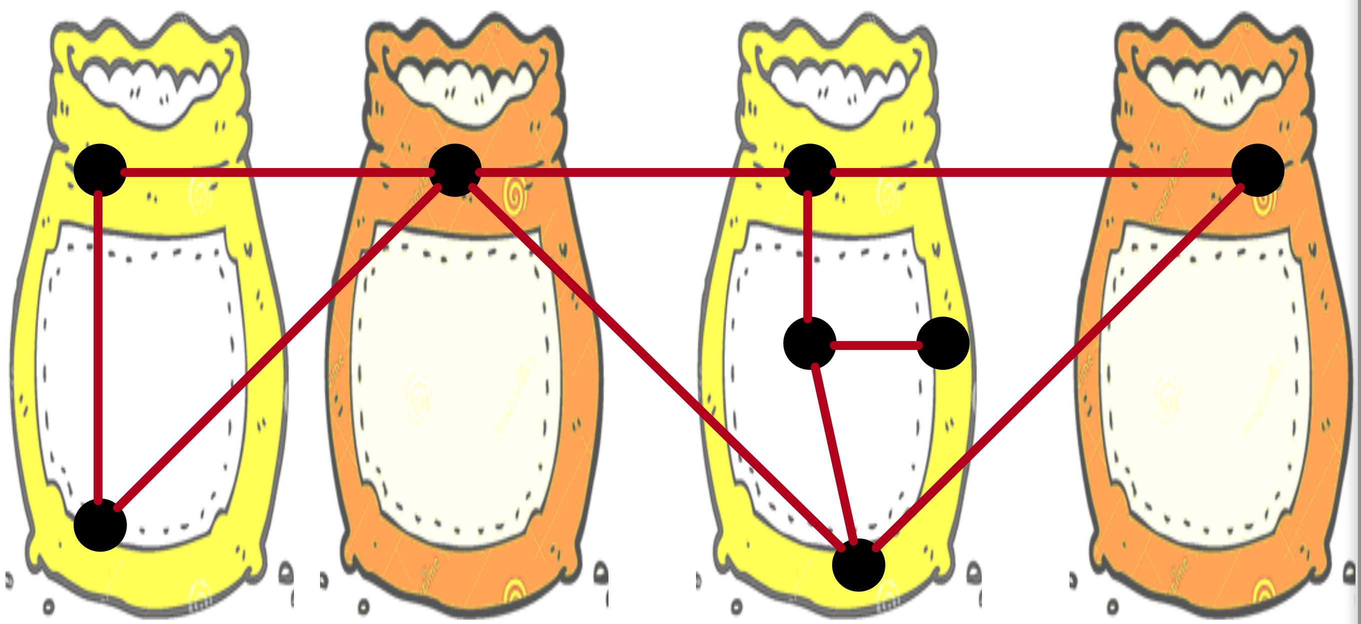
Best known algorithm prior to our result was a very simple algorithm, for a deceptively simple problem.



- For every coloring of V(G) with two colors:
  - Contract each connected component.
  - If the above is a path, store the number of vertices in it.
- Return the **maximum** over the stored numbers.

STARTING POINT OF THE RESULT

Best known algorithm prior to our result was a very simple algorithm, for a deceptively simple problem.



$O^*(2^n)$ algorithm!

- For every coloring of V(G) with two colors:
  - Contract each connected component.
  - If the above is a path, store the number of vertices in it.
- Return the **maximum** over the stored numbers.

OUR RESULT

Can we break the 2^n barrier? YES!

PATH CONTRACTION admits an algorithm running in time c^n, where $c<2$.

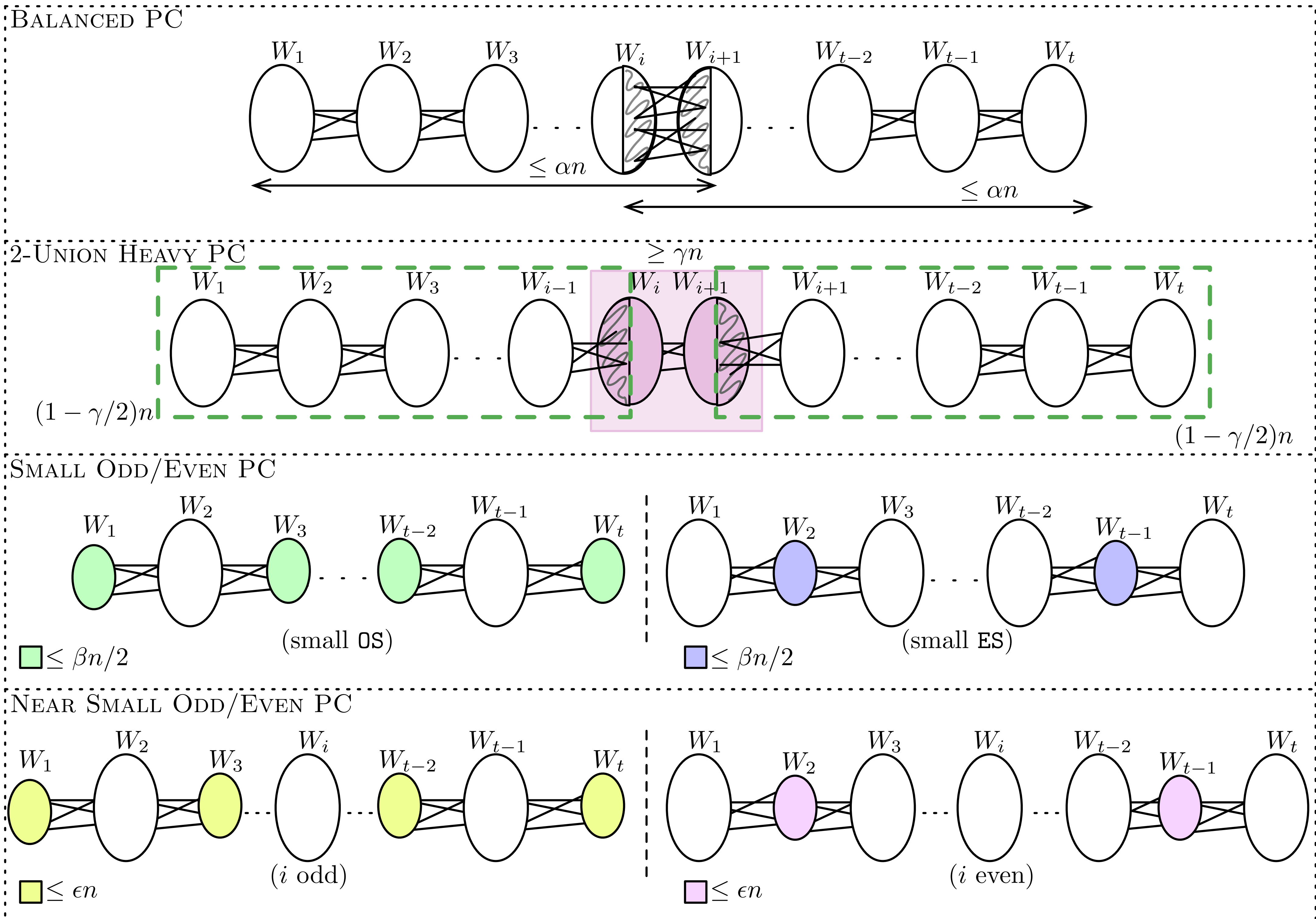
Answers an open question of van't Hof et. al [TCS'09]

OUR METHODS

For a deceptively simple problem, breaking the brute force barrier is quite non-trivial.

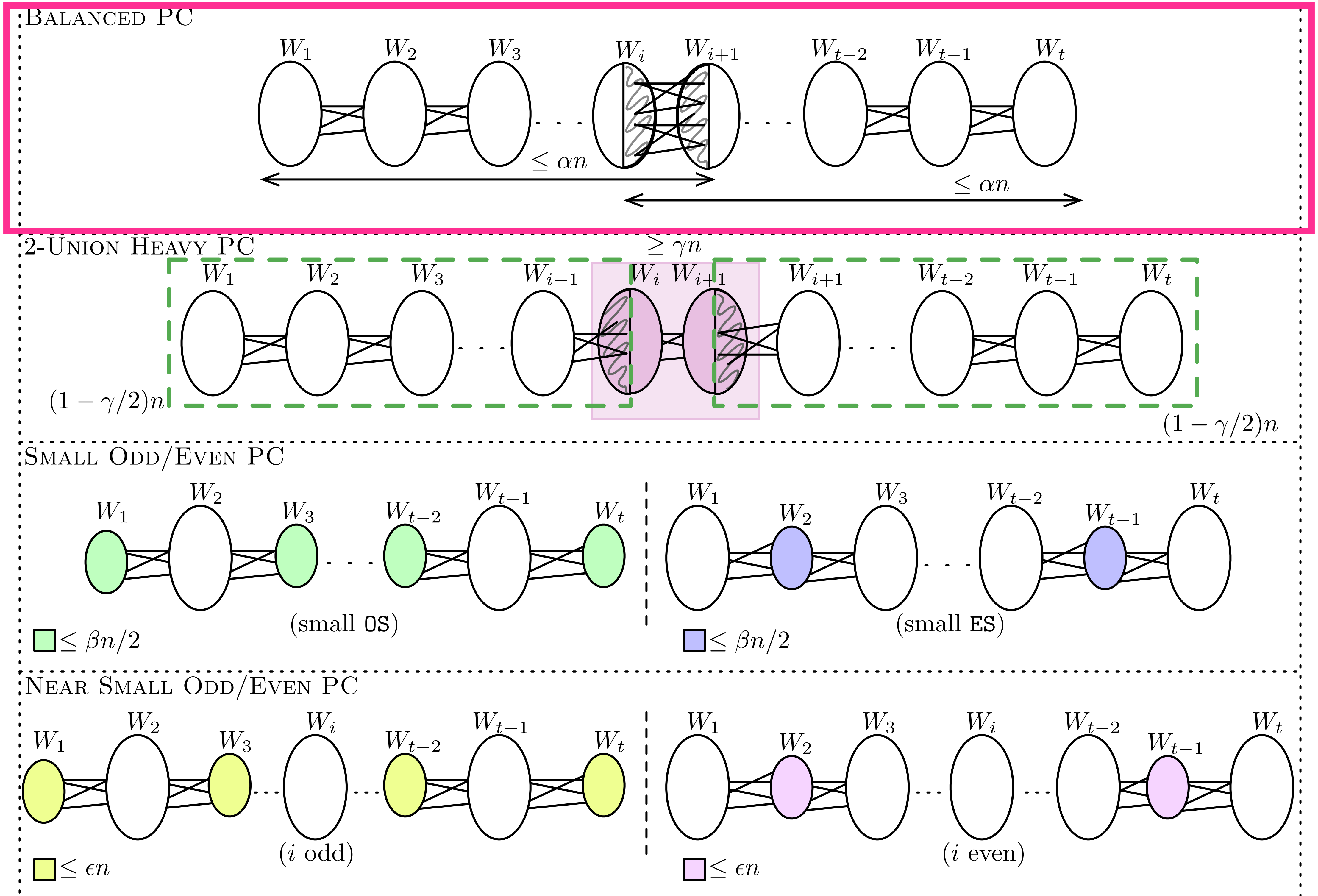
- To obtain our result, we design **four** different algorithms for PATH CONTRACTION, and apply the **most relevant** one for the input.

DIFFRENT ALGORITHMS FOR DIFFERENT CASES



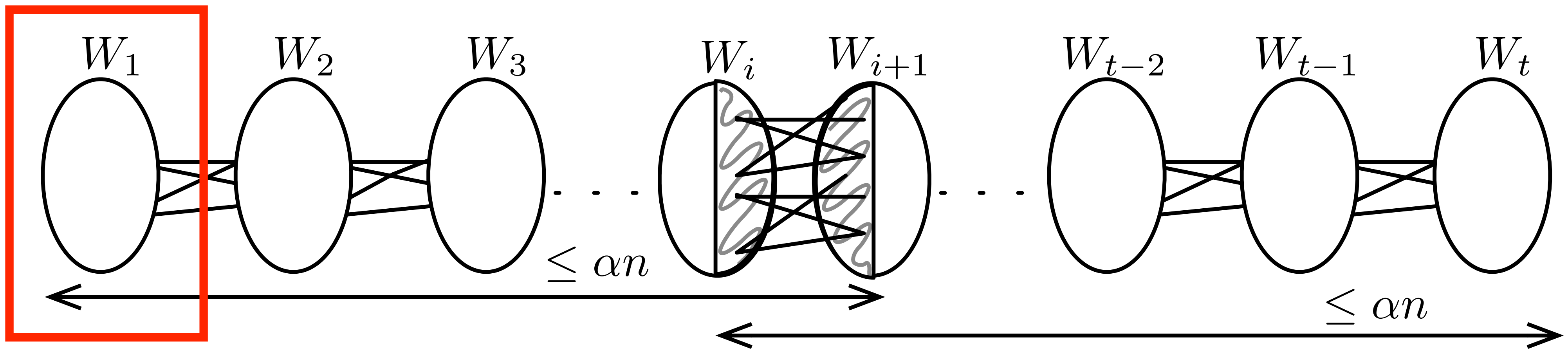
By setting numbers appropriately, one of the algorithms is always relevant for the given input.

DIFFRENT ALGORITHMS FOR DIFFERENT CASES



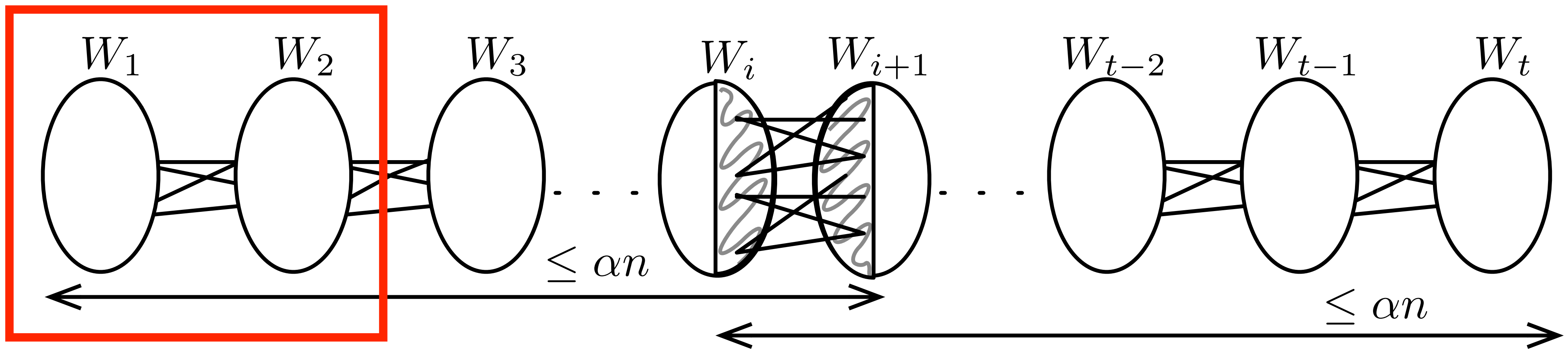
By setting numbers appropriately, one of the algorithms is always relevant for the given input.

BALANCED PC



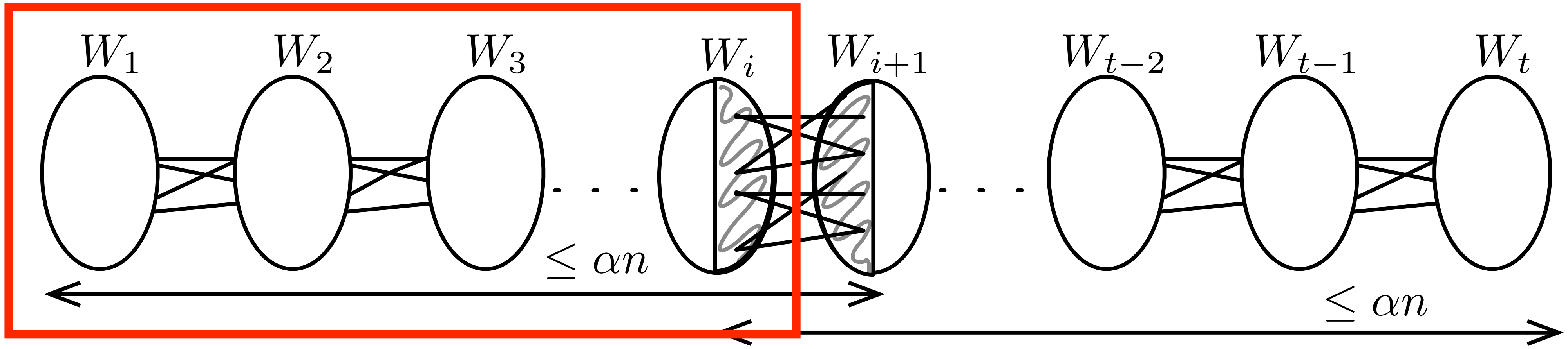
Connected, with small closed neighborhood.

BALANCED PC



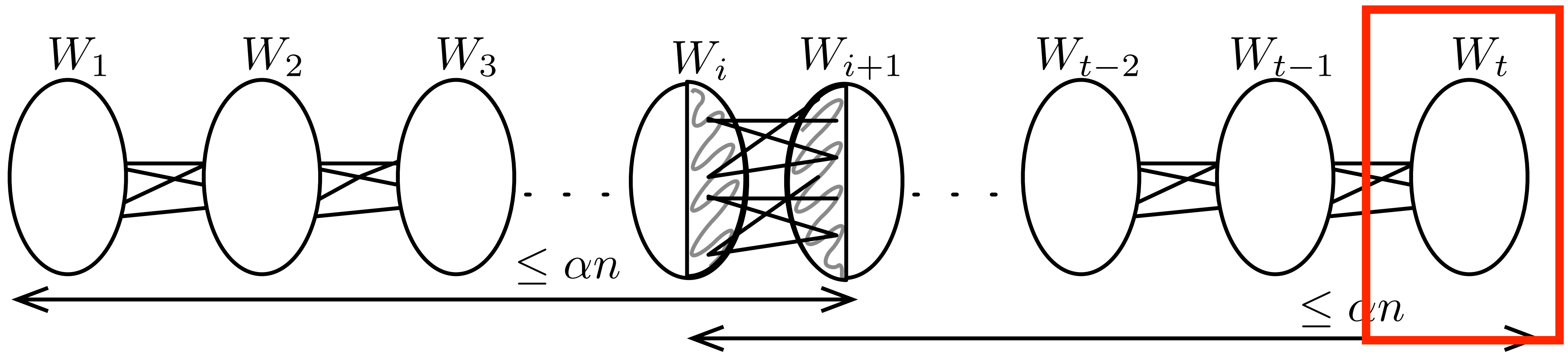
Connected, with small closed neighborhood.

BALANCED PC



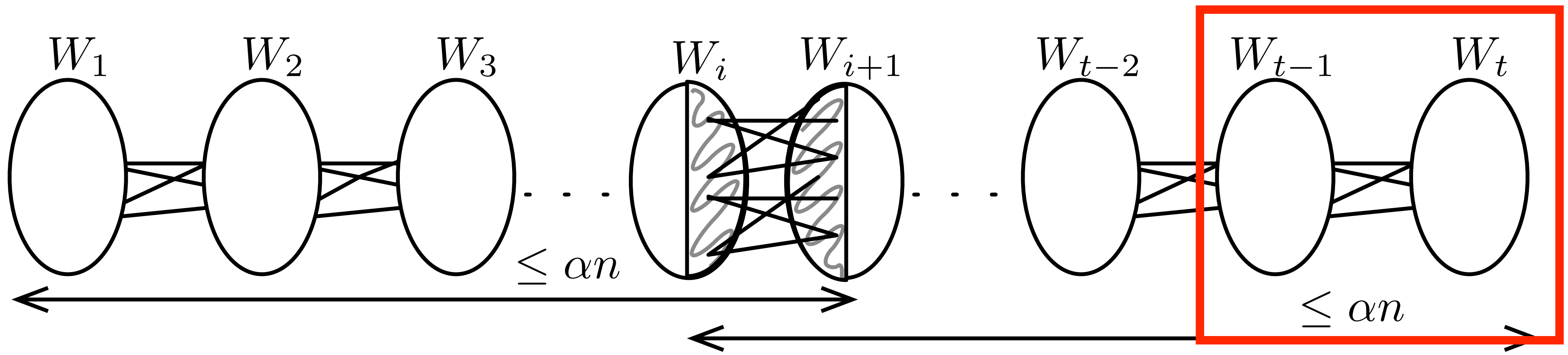
Connected, with small closed neighborhood.

BALANCED PC



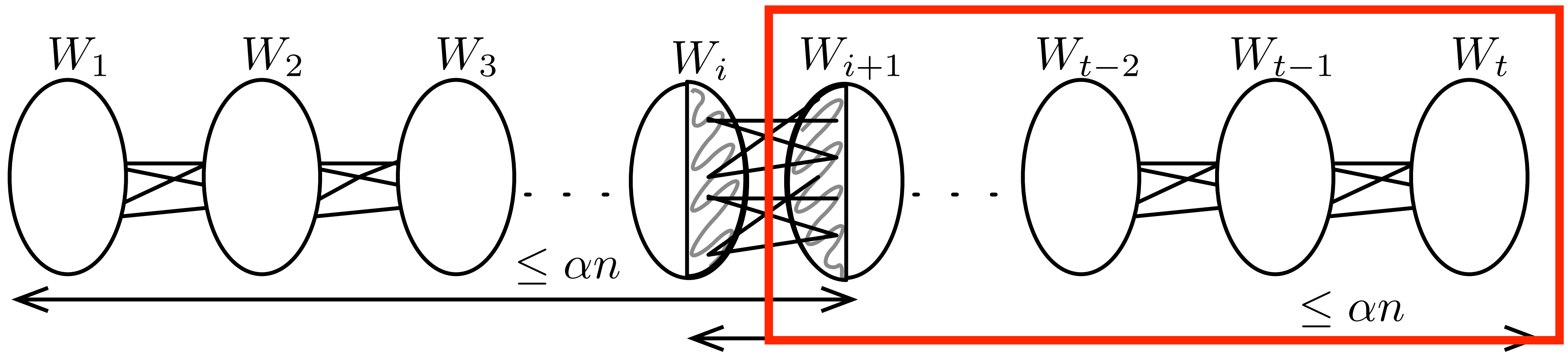
Connected, with small closed neighborhood.

BALANCED PC



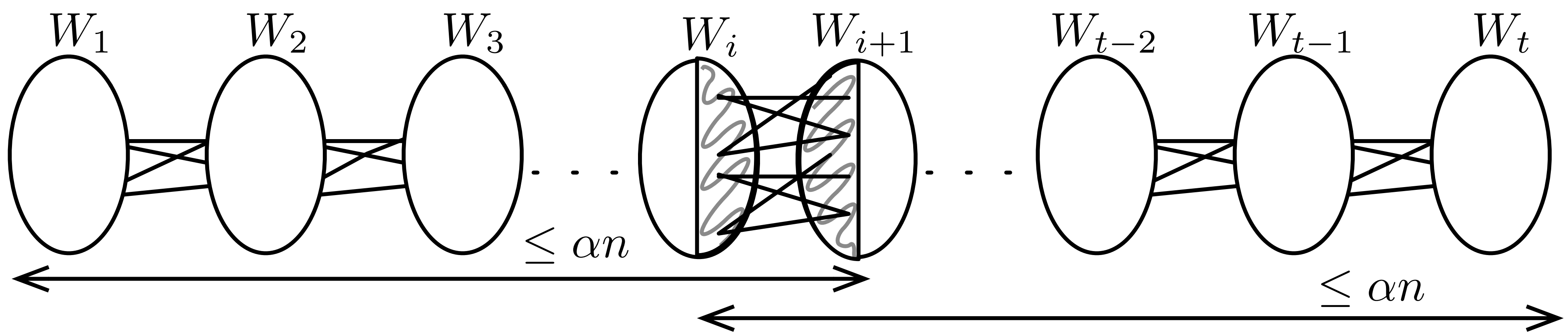
Connected, with small closed neighborhood.

BALANCED PC



Connected, with small closed neighborhood.

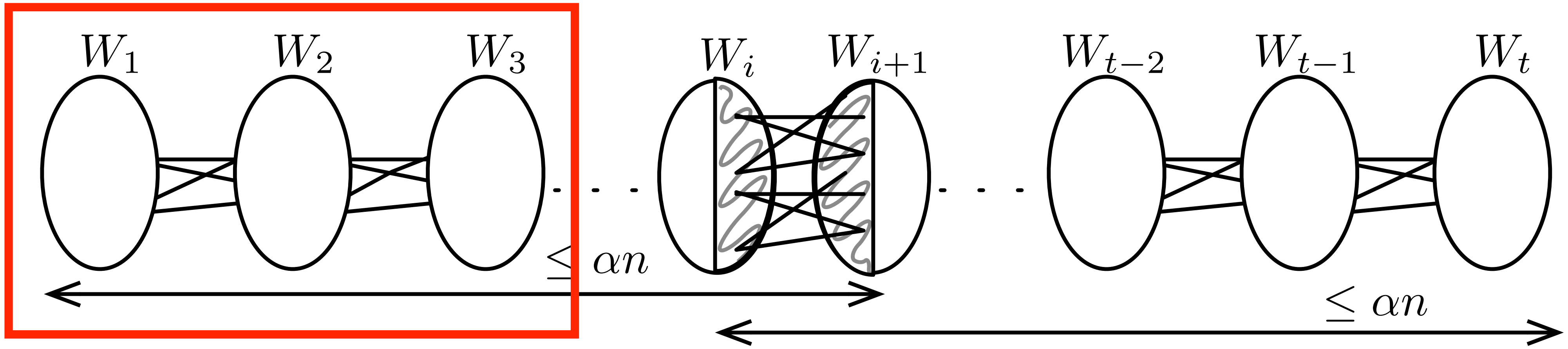
BALANCED PC



- Enumerate all connected sets with small closed neighborhood.

[Better than 2^n]

BALANCED PC



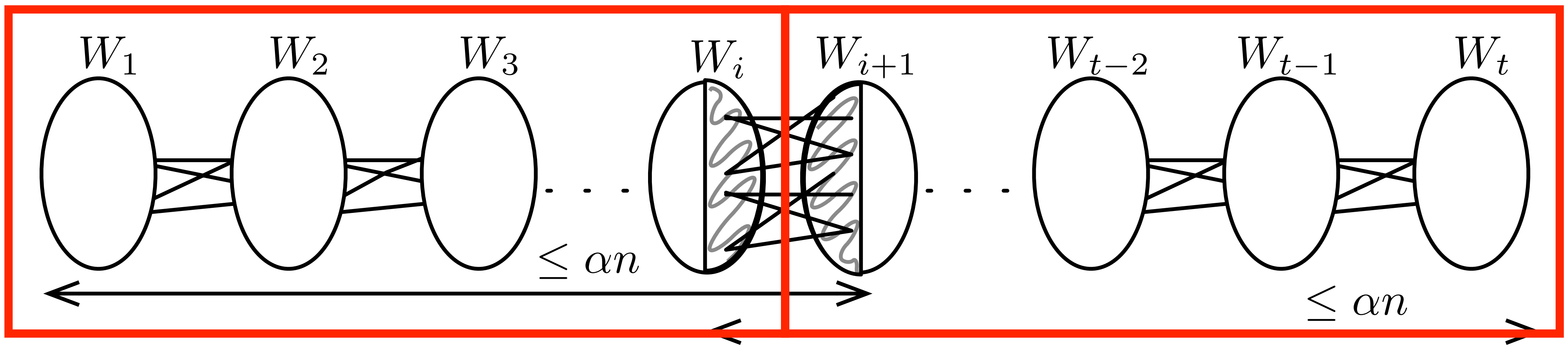
- Enumerate all connected sets with small closed neighborhood.

[Better than 2^n]

- Compute optimal solution for each such set.

[Using Dynamic Programming]

BALANCED PC



- Enumerate all connected sets with small closed neighborhood.

  [Better than 2^n]

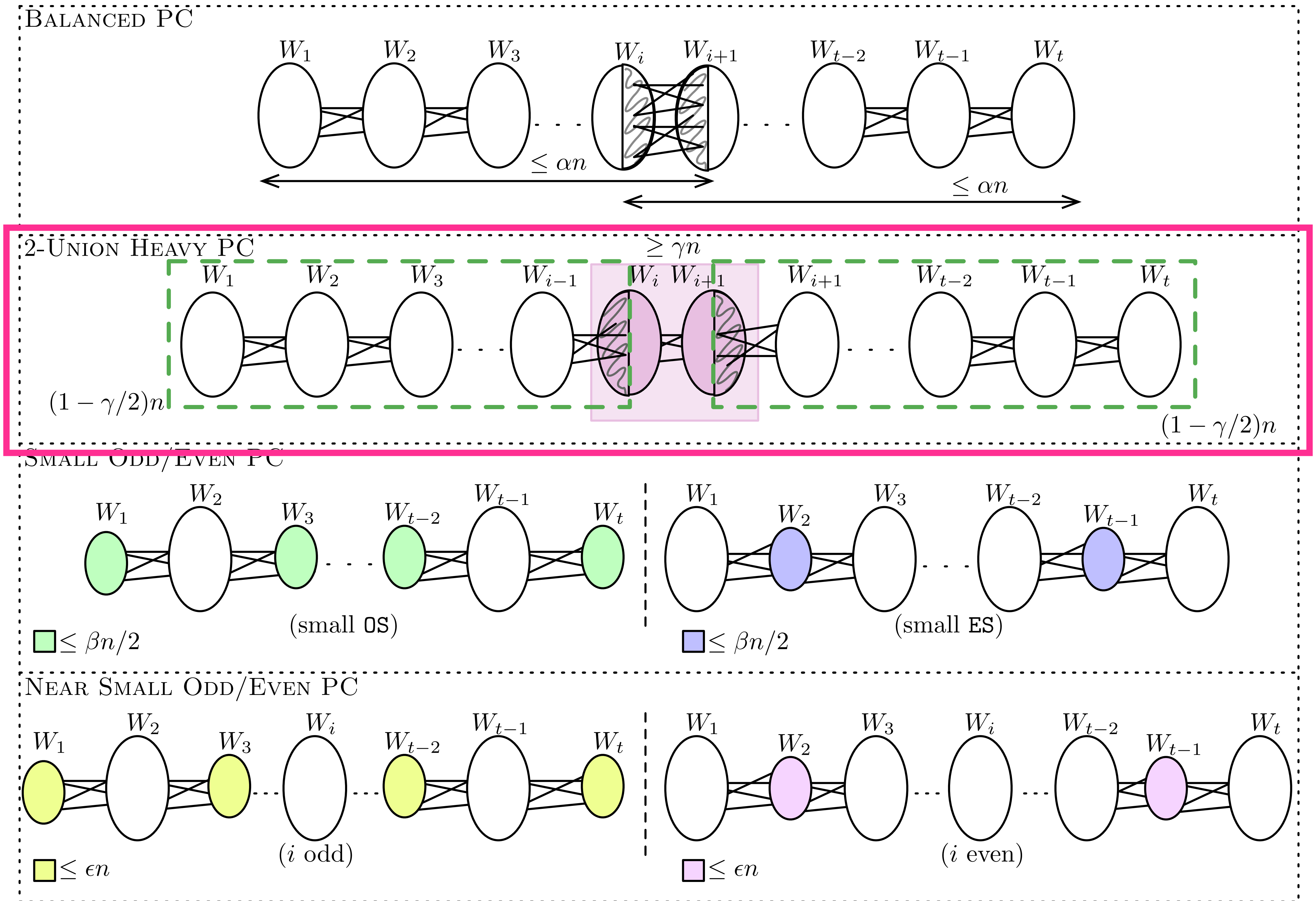
- Compute optimal solution for each such set.

  [Using Dynamic Programming]

- Combine the results.

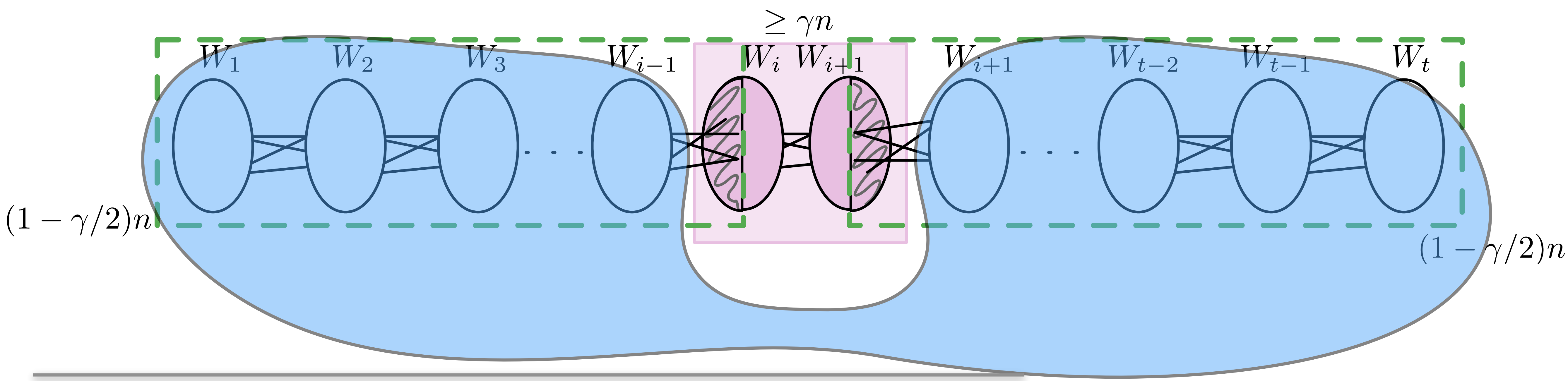
  [For the correct combination, this gives the solution]

DIFFRENT ALGORITHMS FOR DIFFERENT CASES



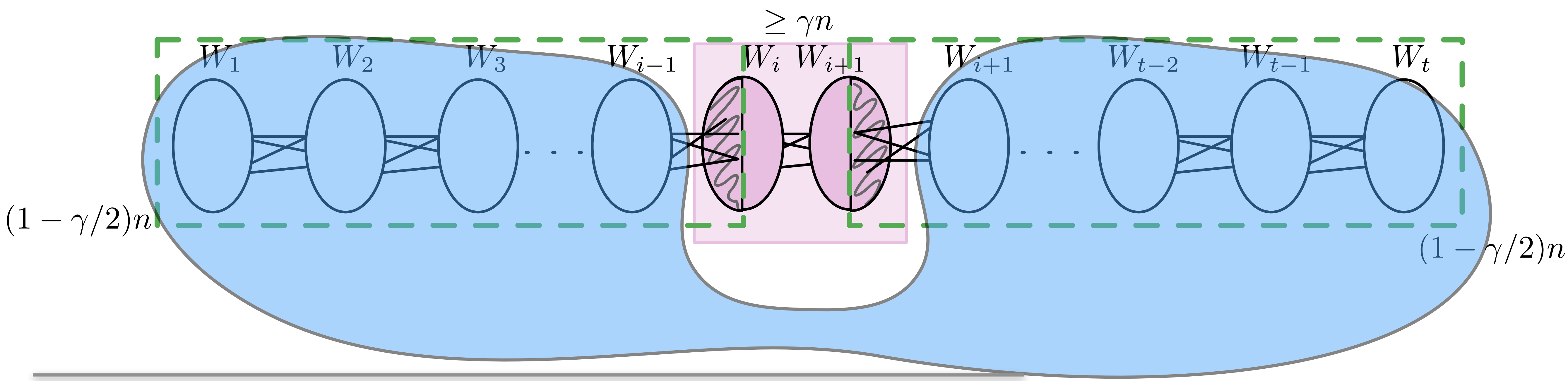
By setting numbers appropriately, one of the algorithms is always relevant for the given input.

2-UNION HEAVY PC



- Find correct blue set, by trying **all possibilities**. Now solve the following partitioning problem.

2-UNION HEAVY PC

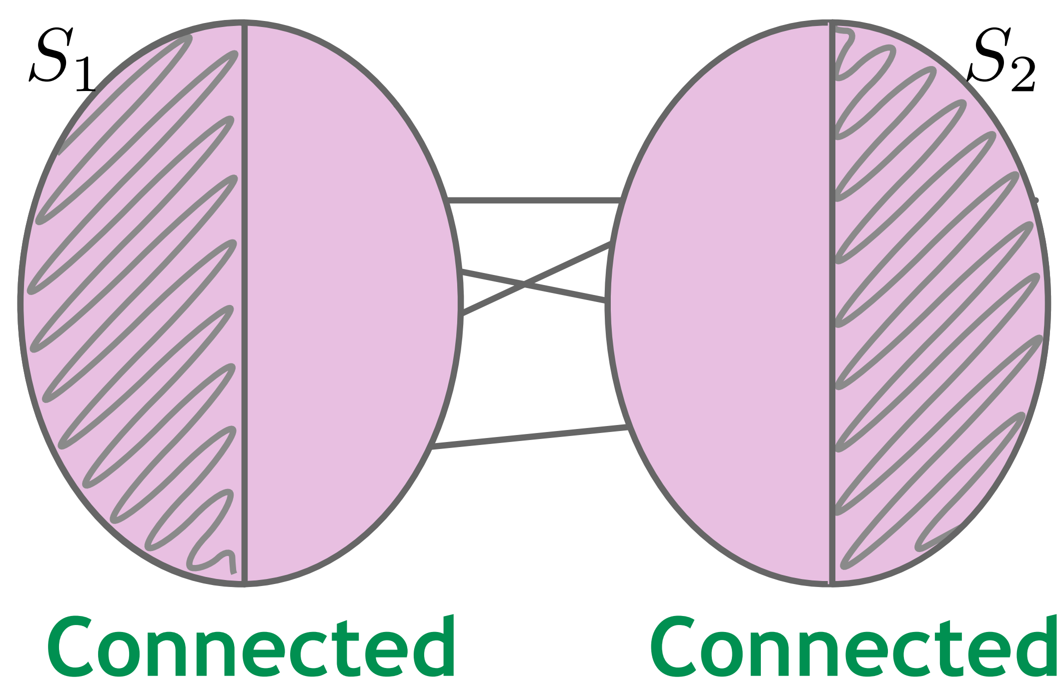


- Find correct blue set, by trying **all possibilities**. Now solve the following partitioning problem.

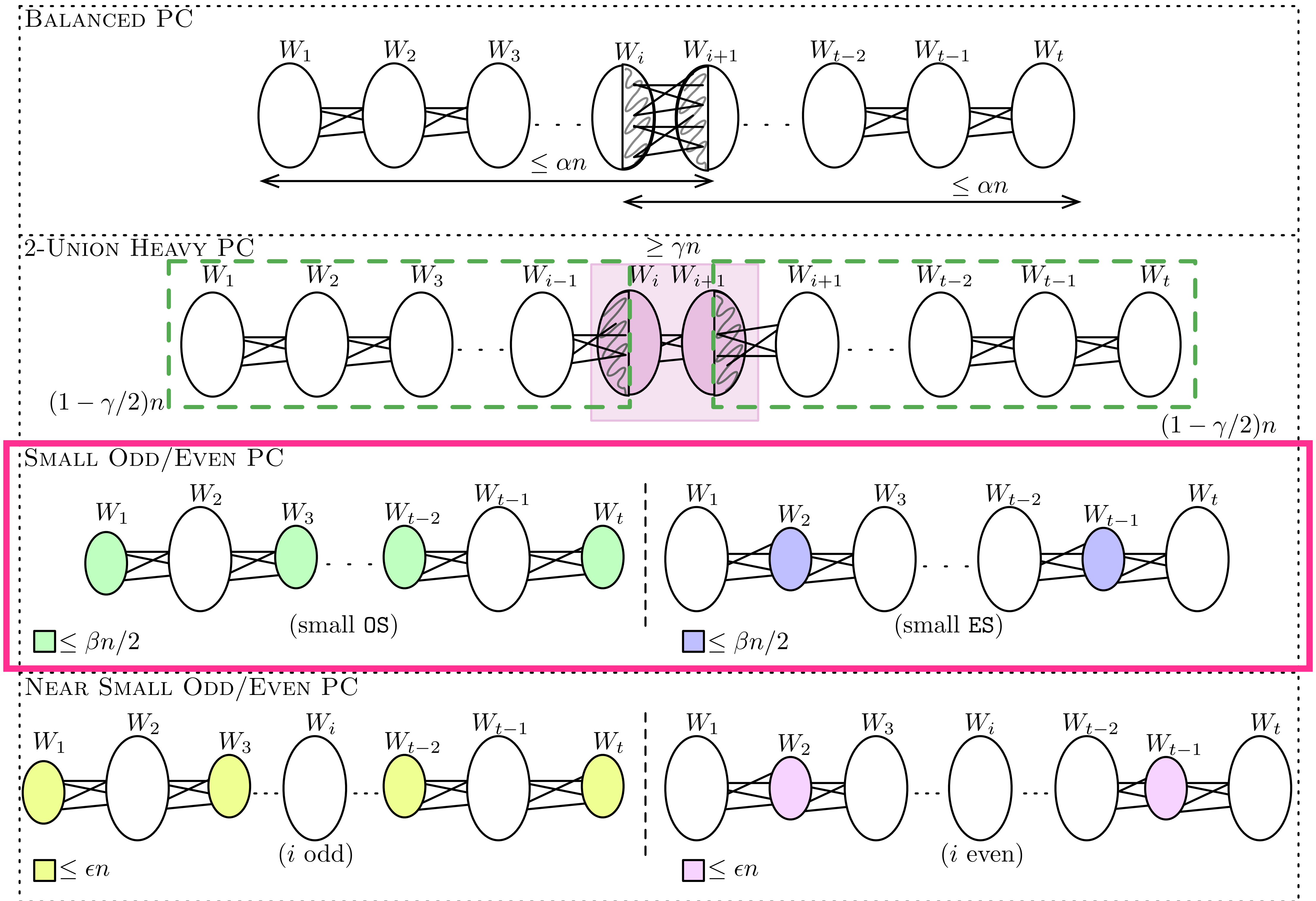
2-DISJOINT CONNECTED SUBGRAPHS

$O^*(1.7804^n)$

[Known]

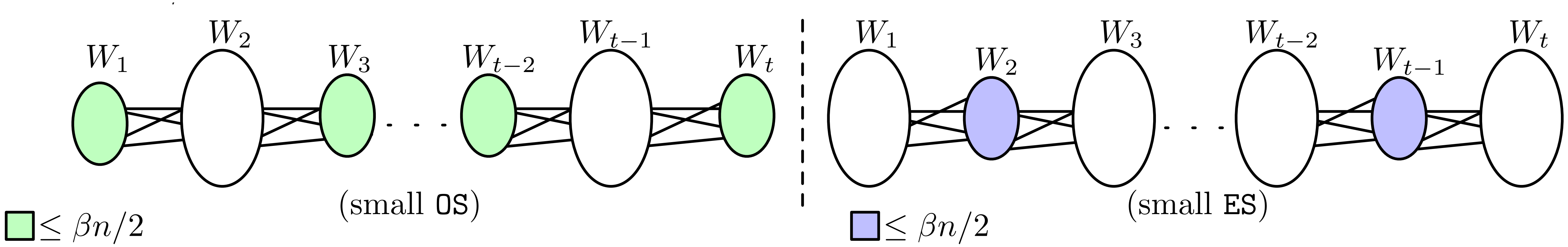


DIFFRENT ALGORITHMS FOR DIFFERENT CASES



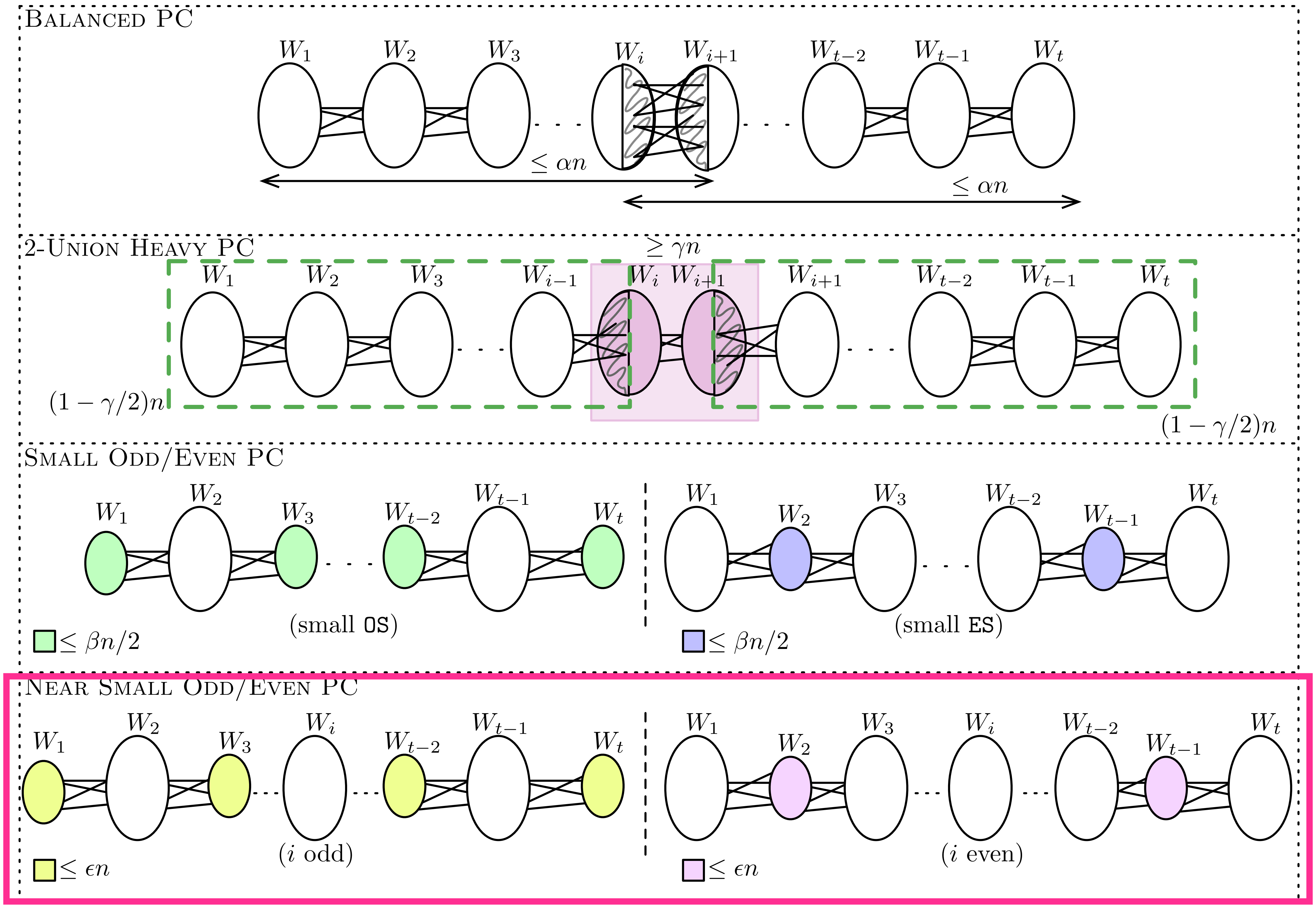
By setting numbers appropriately, one of the algorithms is always relevant for the given input.

SMALL ODD/EVEN PC



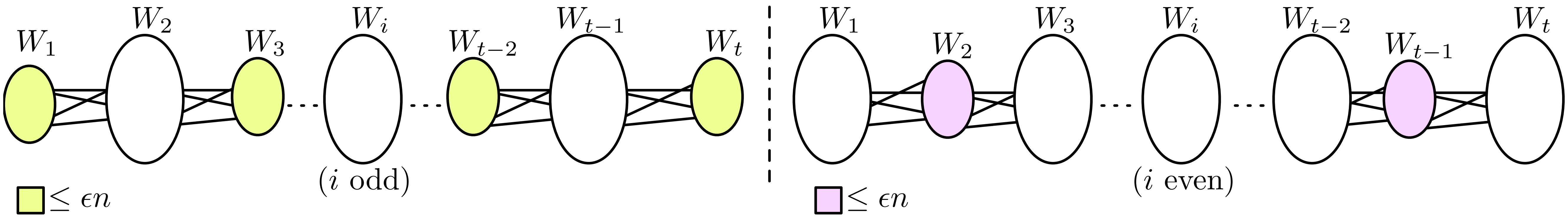
- Find the odd/even set, whichever is smaller, by trying all possibilities.
  - Contract each connected component.
  - If the above is a path, store the number of vertices in it.
- Return the **maximum** over the stored numbers.

DIFFRENT ALGORITHMS FOR DIFFERENT CASES

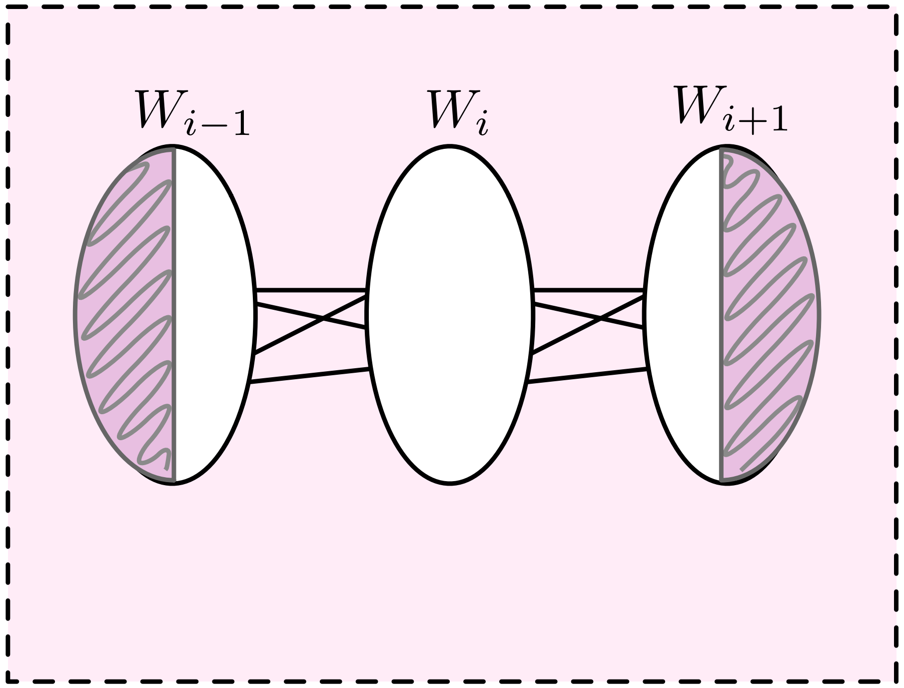


By setting numbers appropriately, one of the algorithms is always relevant for the given input.

NEAR SMALL ODD/EVEN PC

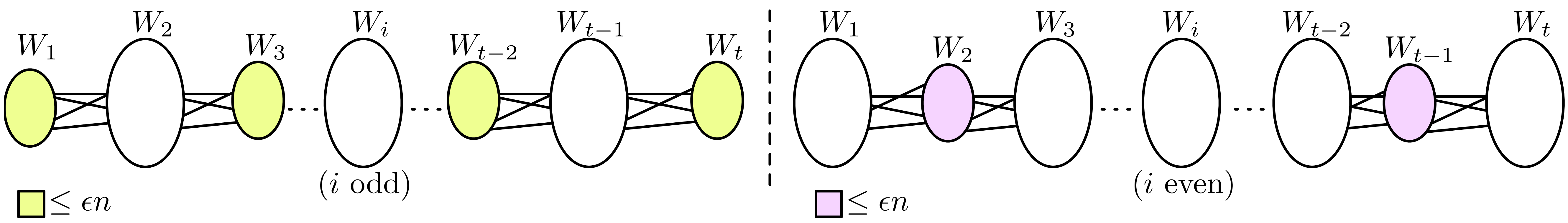


- Find the yellow set, by trying all possibilities.



Large!

NEAR SMALL ODD/EVEN PC

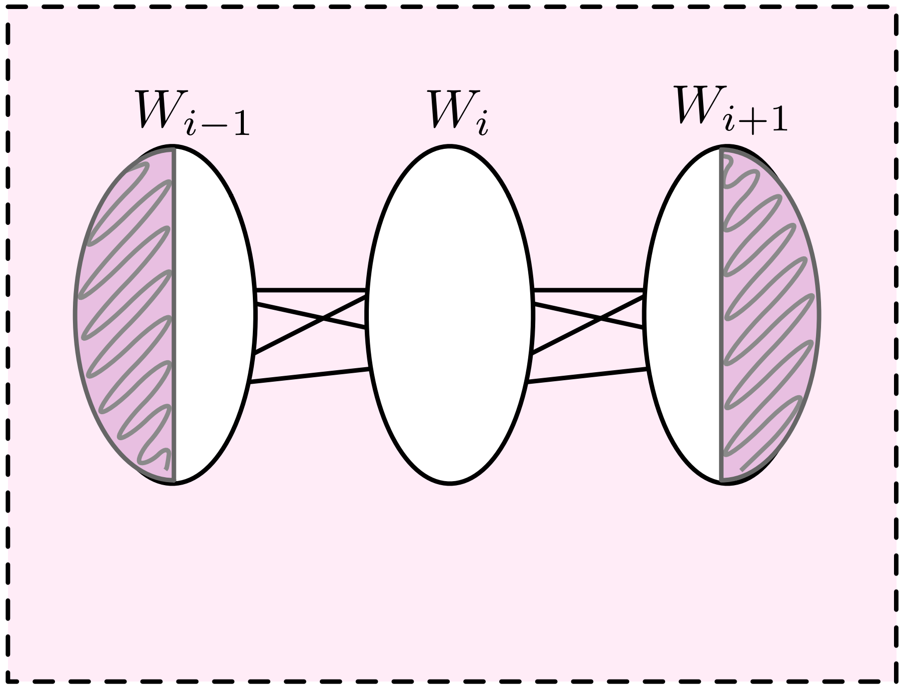


- Find the yellow set, by trying all possibilities.

3-DISJOINT CONNECTED SUBGRAPHS

$O^*(1.88^n)$

[This paper]



Large!

CONCLUSION

- We obtained an algorithm that breaks the 2^n barrier for PATH CONTRACTION, where the improvement is very marginal.
- We gave an $O^*(1.88^n)$ algorithm for 3-DISJOINT CONNECTED SUBGRAPHS, which is a generalisation of 2-DISJOINT CONNECTED SUBGRAPHS.
- Can we obtain substantially better algorithm for PATH CONTRACTION?

CONCLUSION

- We obtained an algorithm that breaks the 2^n barrier for PATH CONTRACTION, where the improvement is very marginal.
- We gave an $O^*(1.88^n)$ algorithm for 3-DISJOINT CONNECTED SUBGRAPHS, which is a generalisation of 2-DISJOINT CONNECTED SUBGRAPHS.
- Can we obtain substantially better algorithm for PATH CONTRACTION?

Thanks!