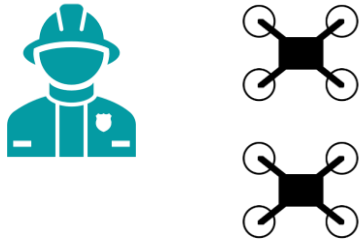
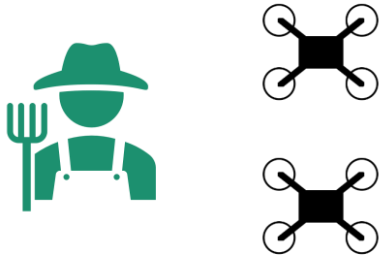
Tournament Selection Algorithm for the Multiple Travelling Salesman Problem

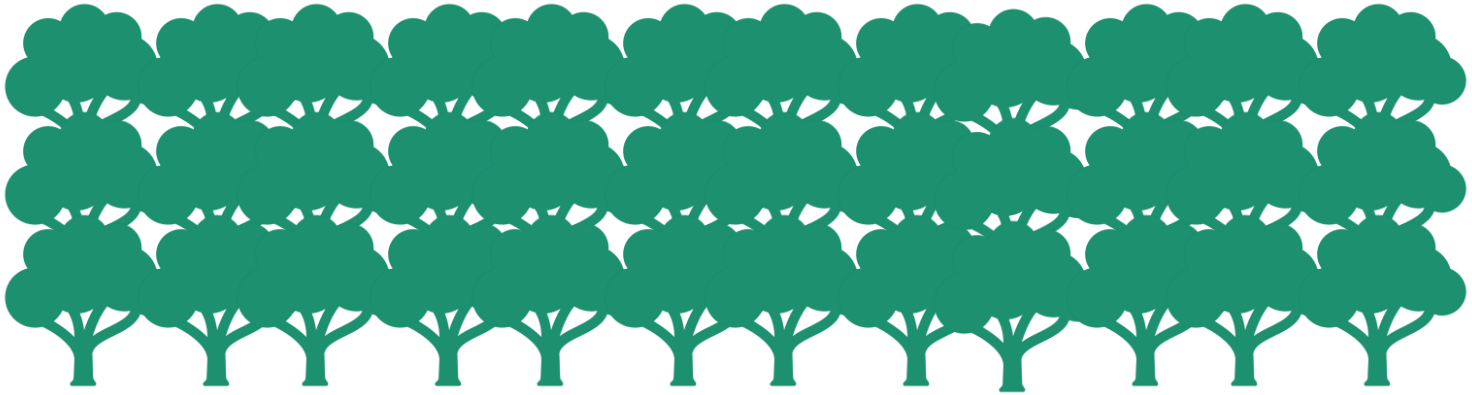
Giorgos Polychronis and Spyros Lalis
Electrical and Computer Engineering Dept.
University of Thessaly
Volos, Greece

- Motivation
- Problem formulation
- Approach
- Experiments and results
- Conclusion

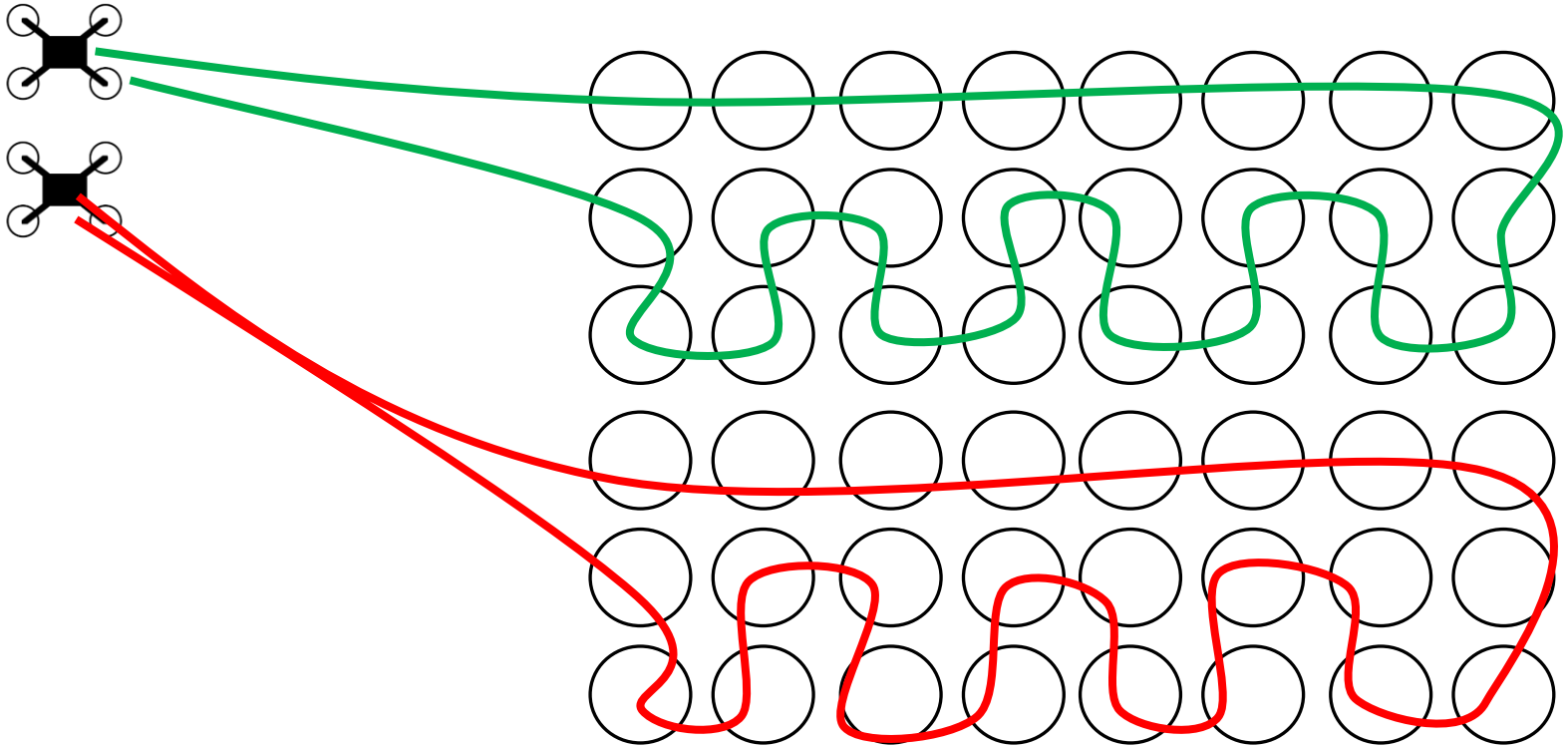
Application Scenarios







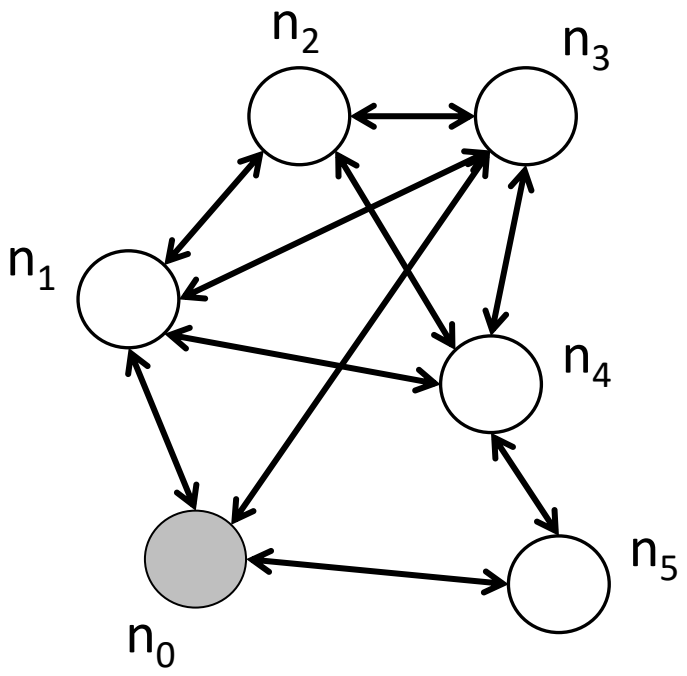
The problem



- Motivation
- Problem formulation
- Approach
- Experiments and results
- Conclusion

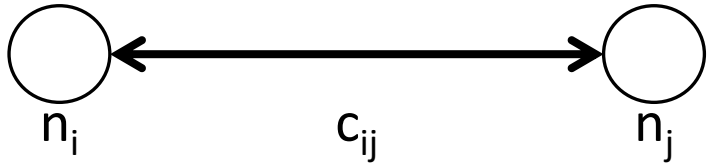
Terrain

- G = (N,E)
- N is the set of nodes
- n_0 is the depot node
- E is the set of edges



Edge cost

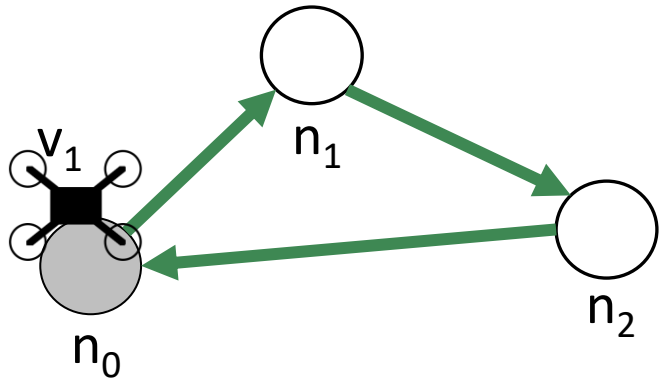
- Let $c_{ij} > 0$, be the cost of edge e_{ij} , c_{ij} corresponding to the time needed to cross e_{ij}



Vehicle makespan

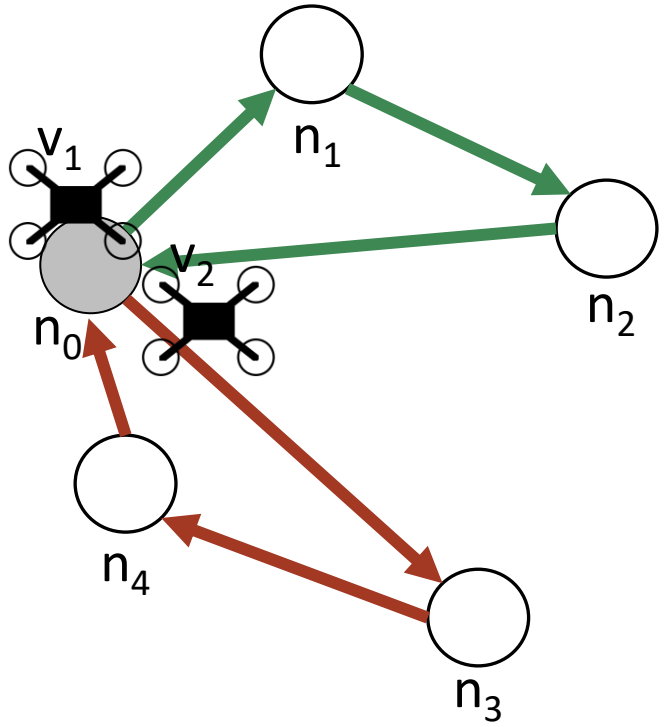
- Let the makespan of one vehicle be the sum of the costs of the edges in its route

- $\text{makespan}_{v1} = c_{0,1} + c_{1,2} + c_{2,0}$



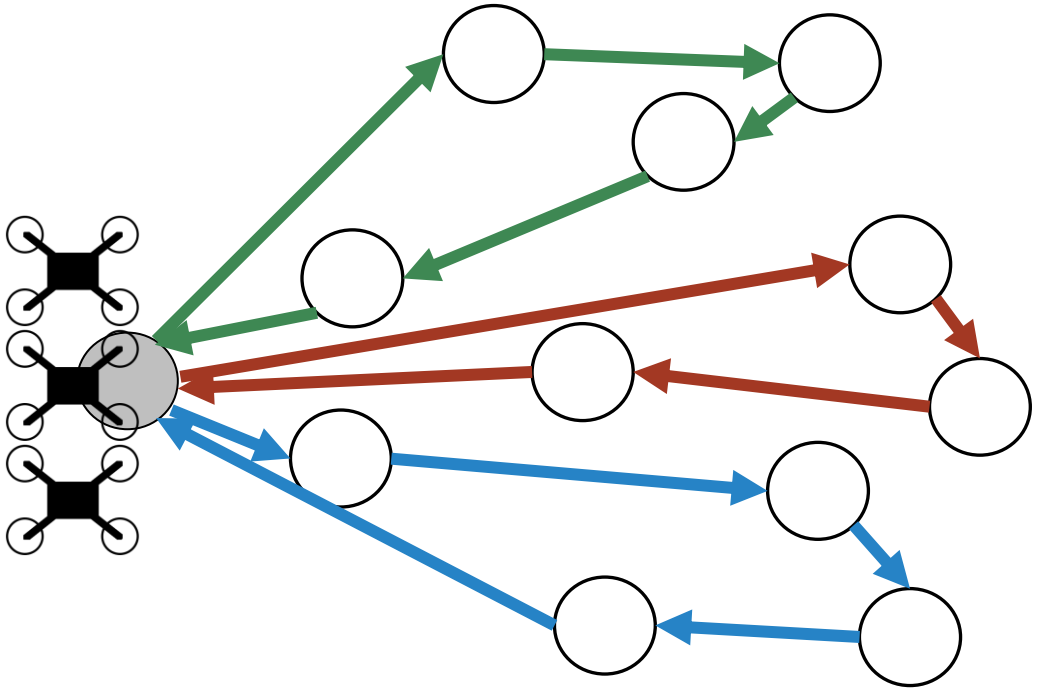
Mission makespan

- Let the mission's makespan be the maximum makespan of all vehicles
- makespan = $\max(\text{makespan}_{v1}, \text{makespan}_{v2}) = \max(c_{0,1}+c_{1,2}+c_{2,0}, c_{0,3}+c_{3,4}+c_{4,0})$



Objective

- Assume M homogeneous vehicles
- Find a route for each vehicle such that the mission's makespan is minimized



- Motivation
- Problem formulation
- Approach
- Experiments and results
- Conclusion

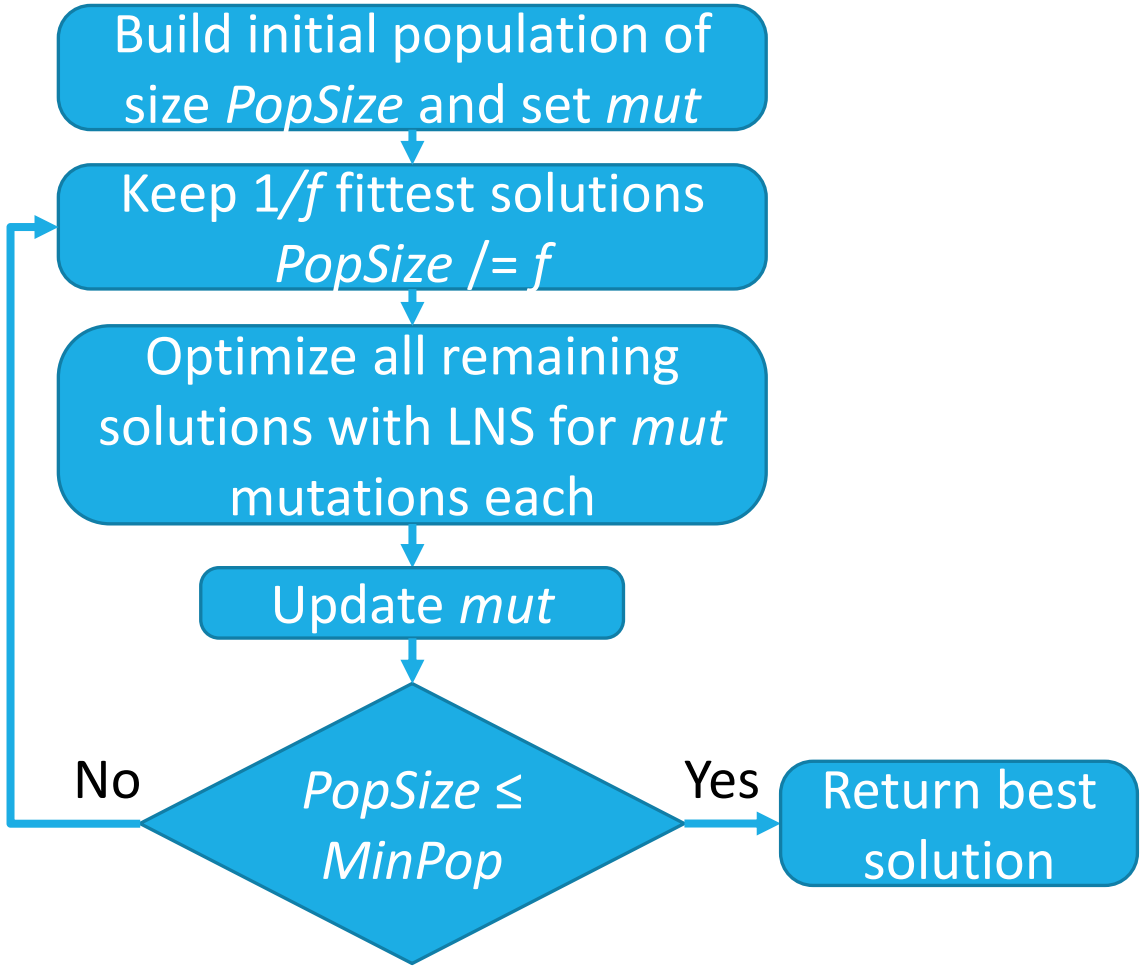
Main loop

Combination of:

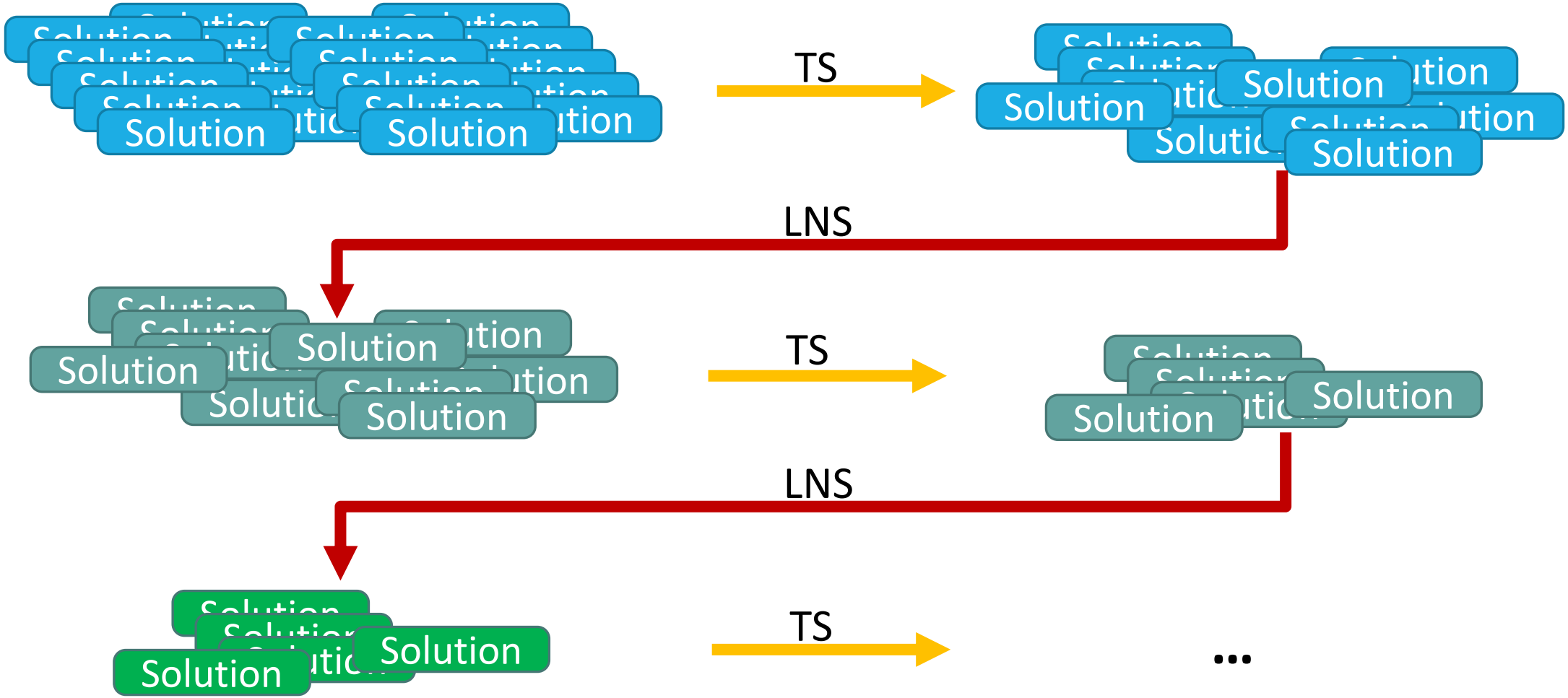
- Tournament Selection (TS)

and

- Large Neighborhood Search (LNS)

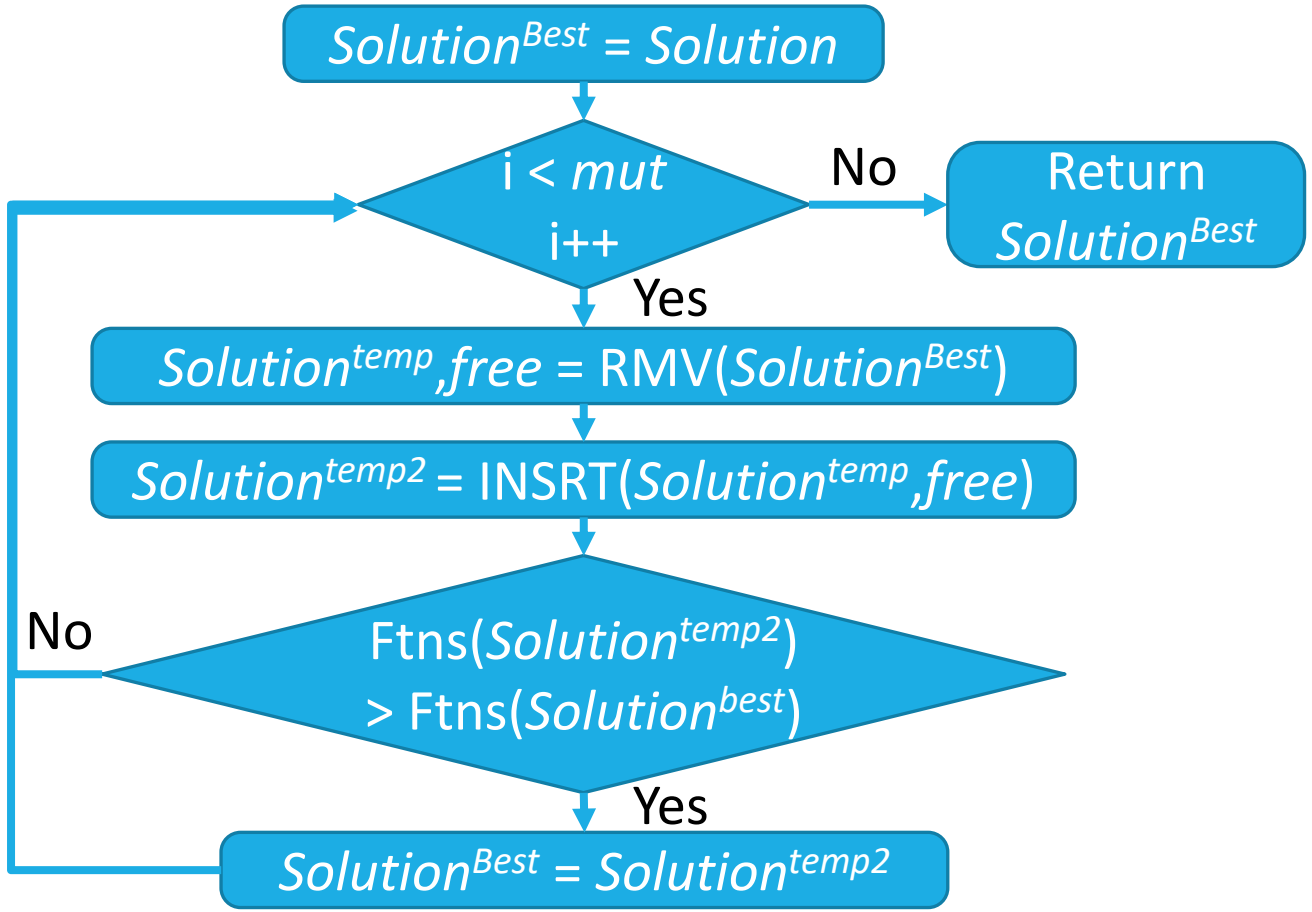


Evolution of population

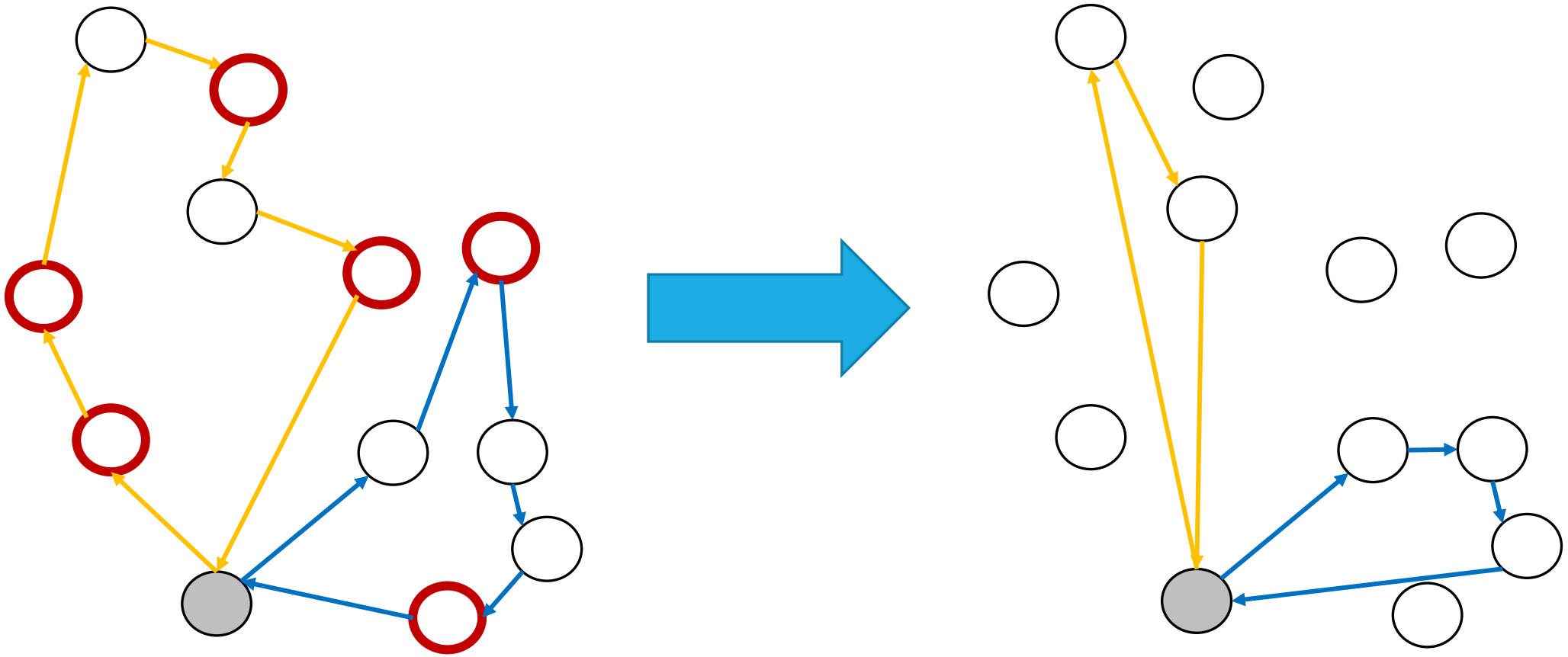


Large Neighborhood Search (LNS)

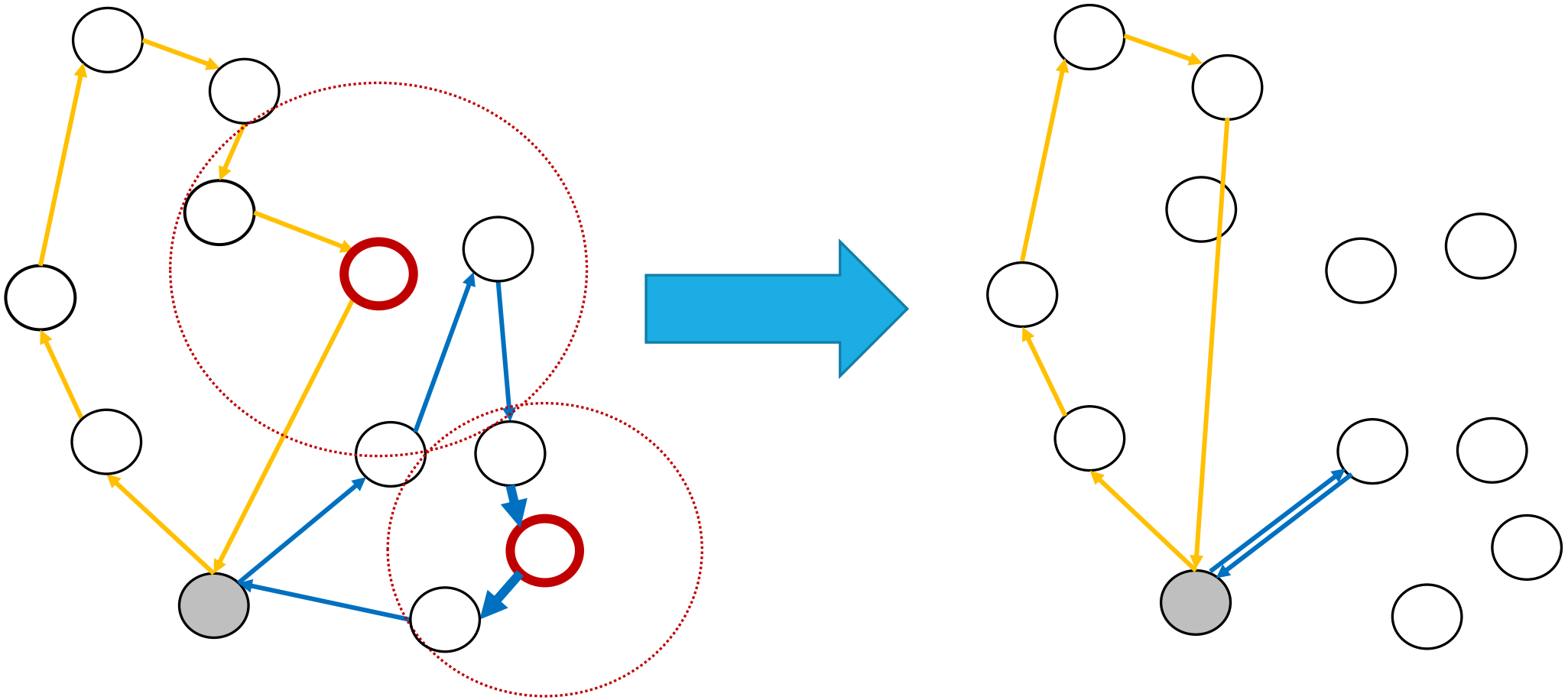
- Step 1: removes some random nodes from the best so far solution
- Step 2: reinserts the nodes removed back to the solution



Remove random



Remove proximity



Node removal parameters

Random node removal

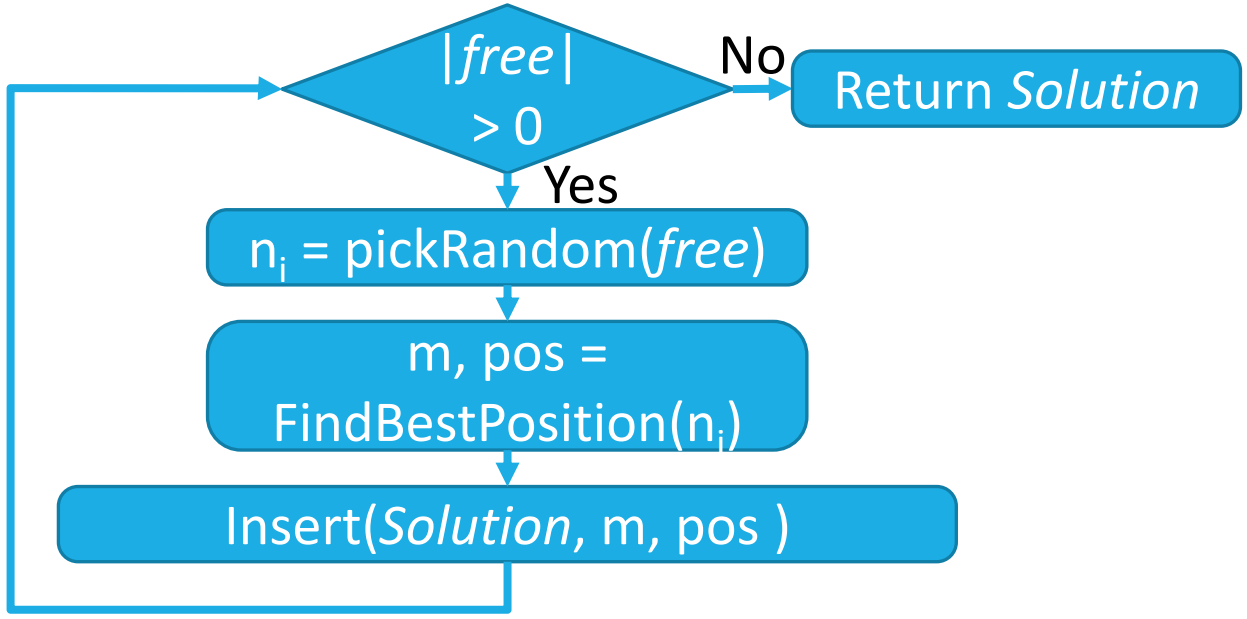
- Remove k nodes
- $k \in [0.2*N \,..\, 0.4*N]$

Proximity-based node removal

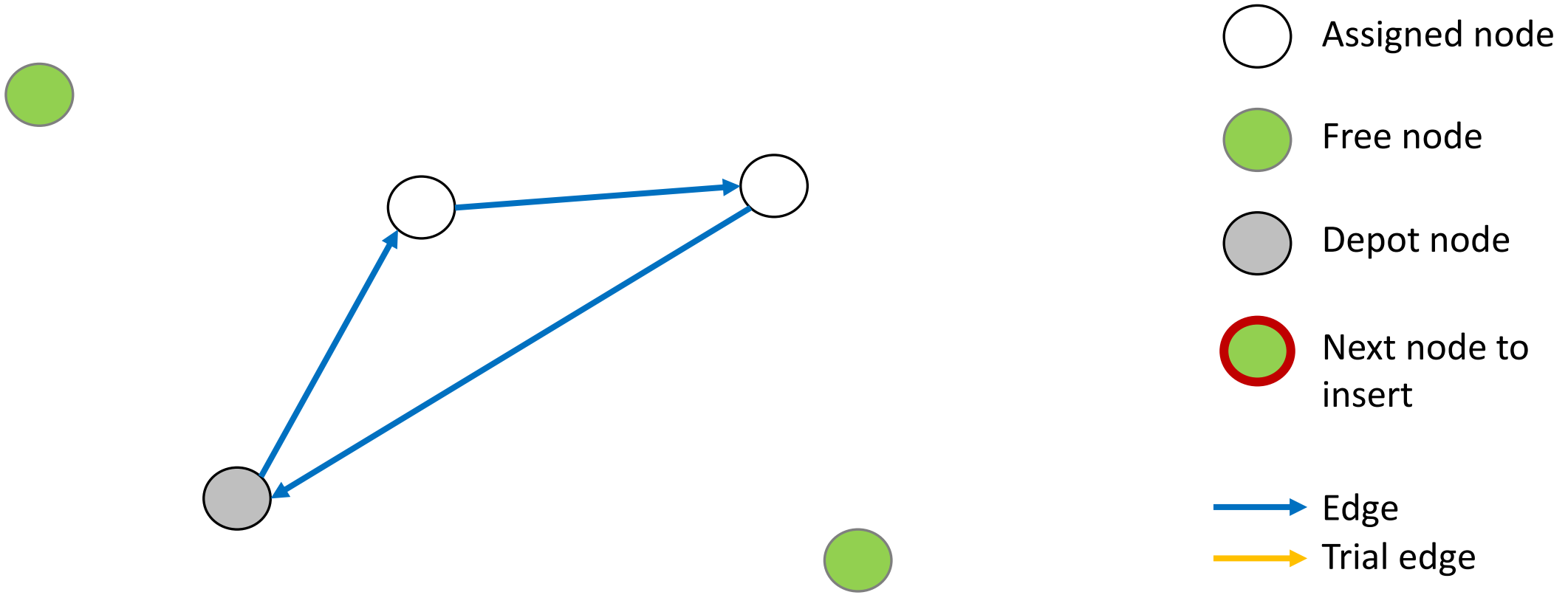
- Remove a total of k nodes
- $k \in [\sqrt{N} \,..\, 4*\sqrt{N}]$
- Around s seeds
- $s \in [1 \,..\, 0.4*\sqrt{N}]$

Node insertion

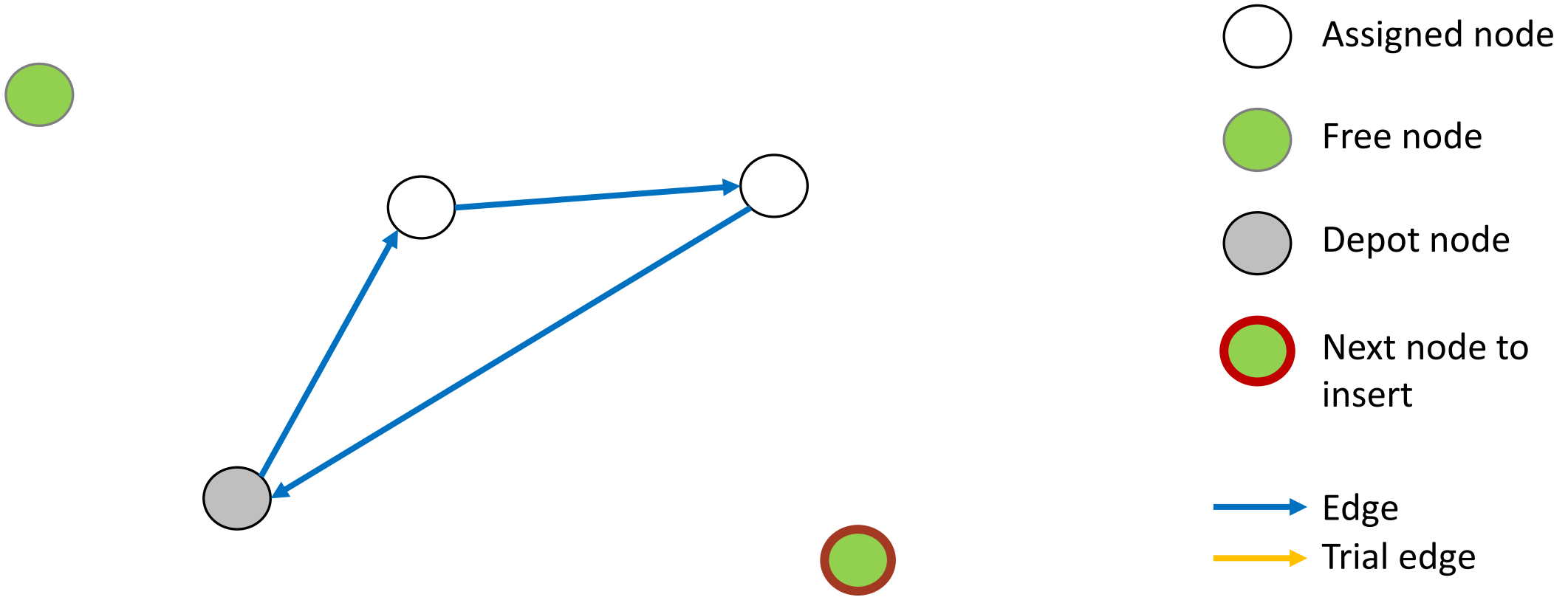
- Picks a random node from the free list
- Inserts the node in the best position



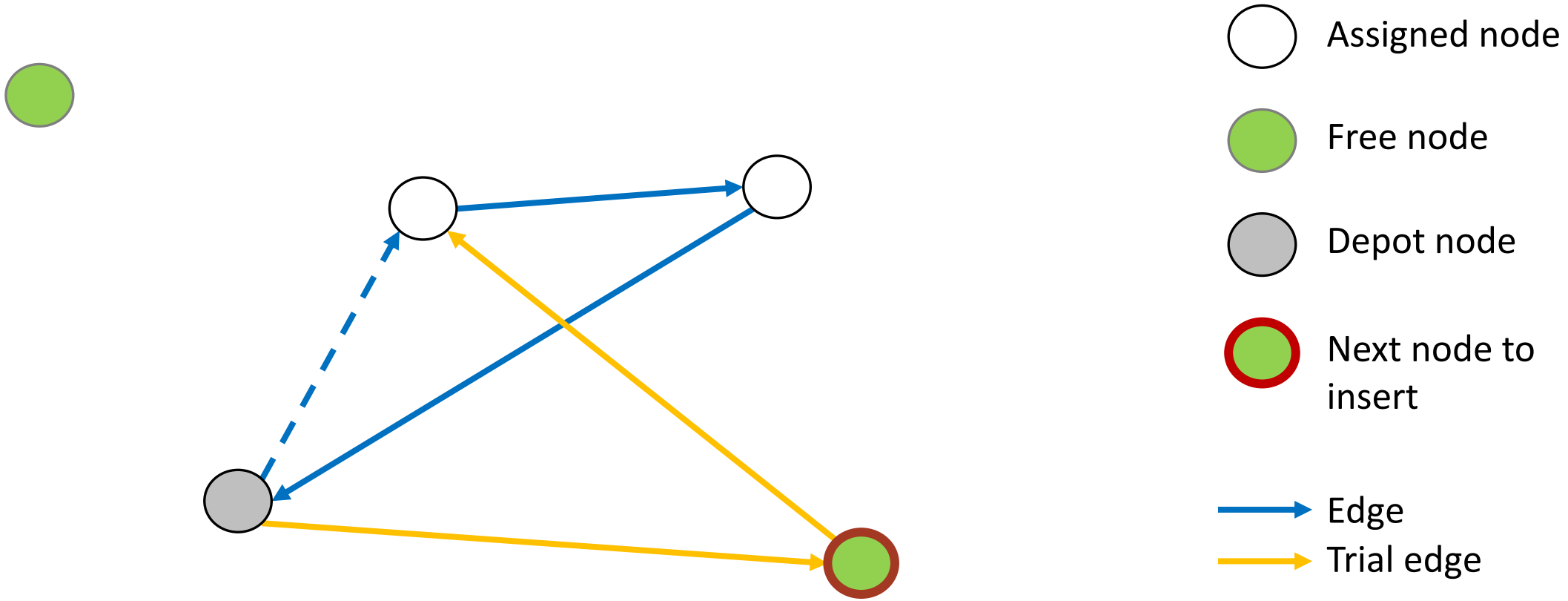
Insertion example 1 (1/12)



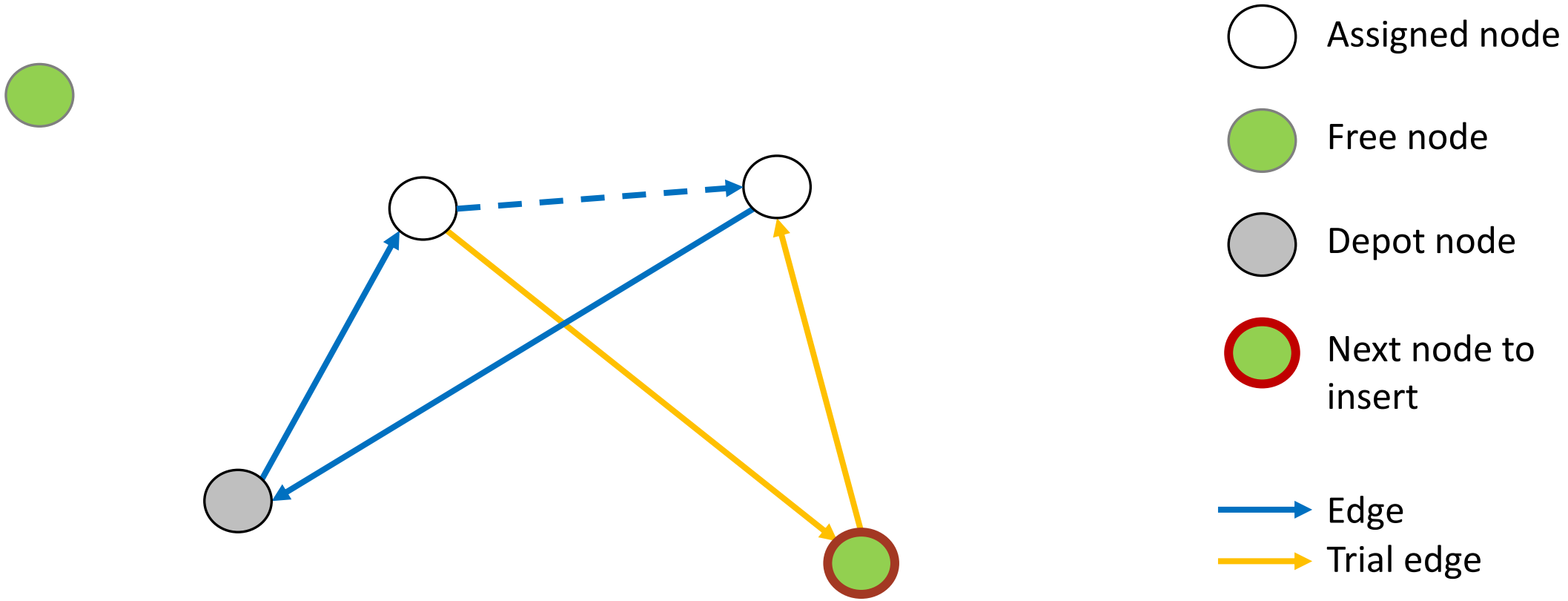
Insertion example 1 (2/12)



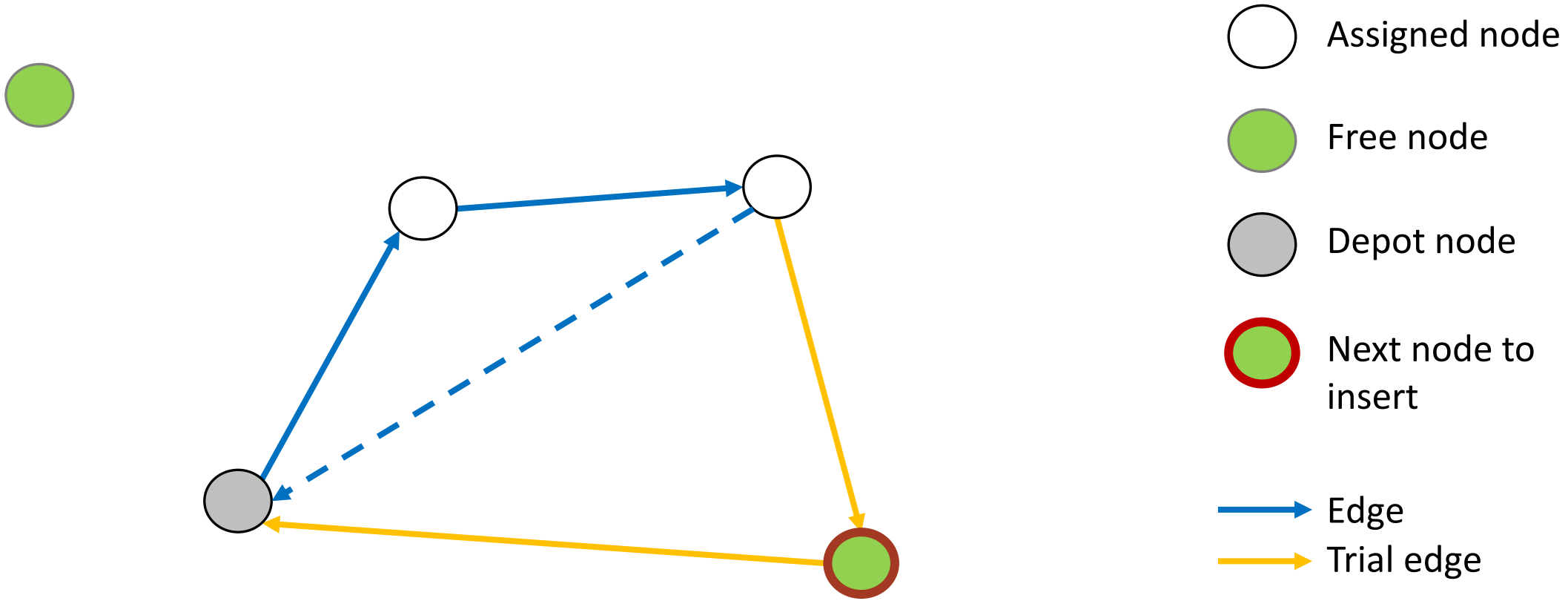
Insertion example 1 (3/12)



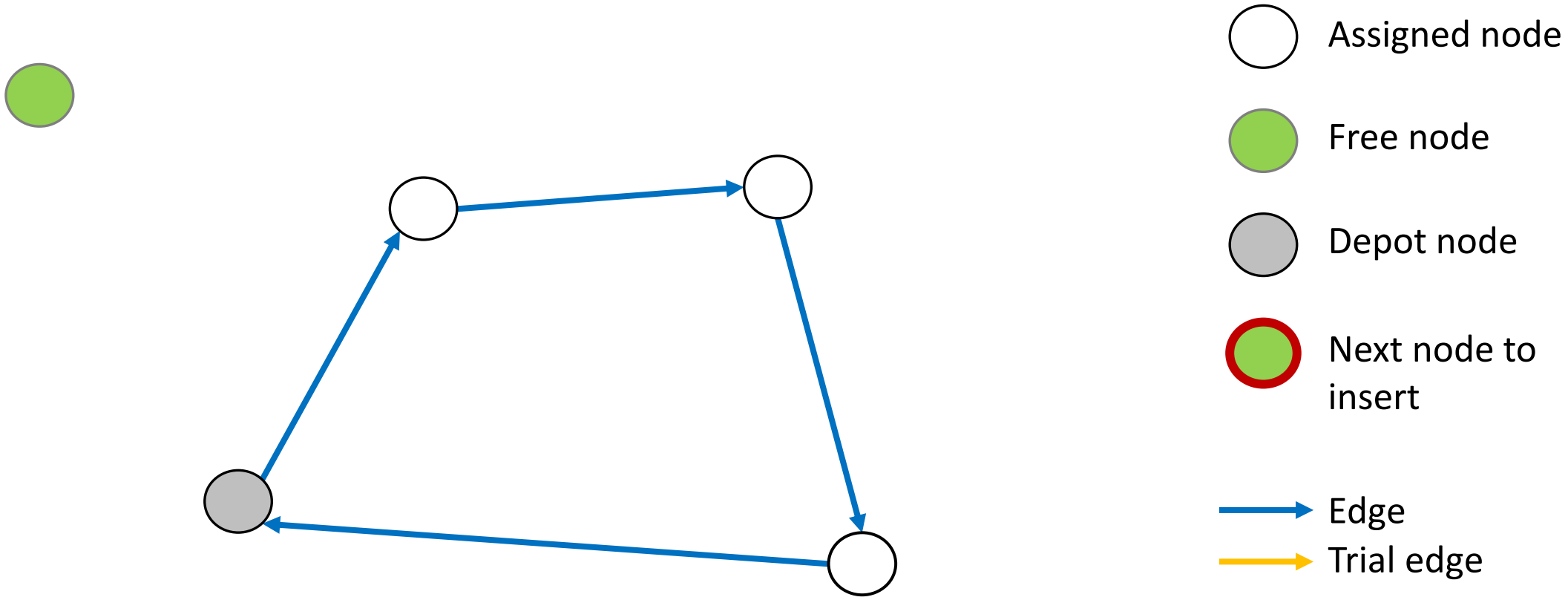
Insertion example 1 (4/12)



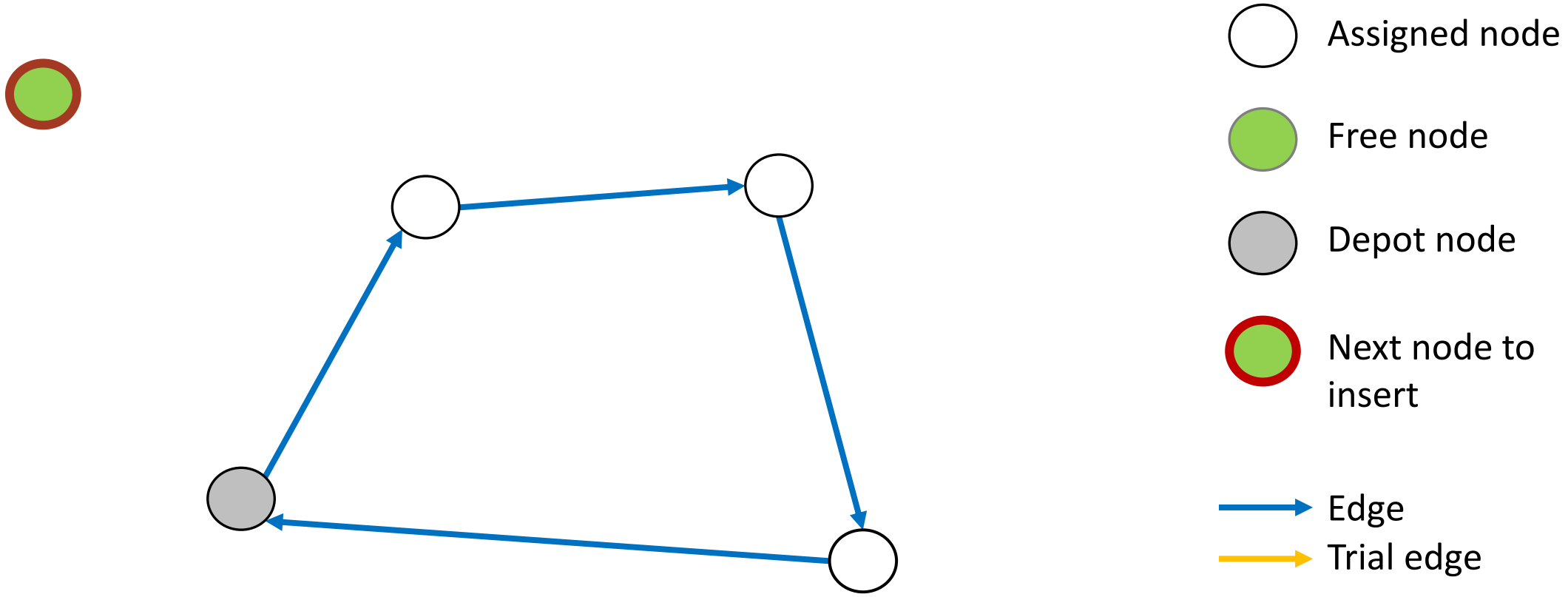
Insertion example 1 (5/12)



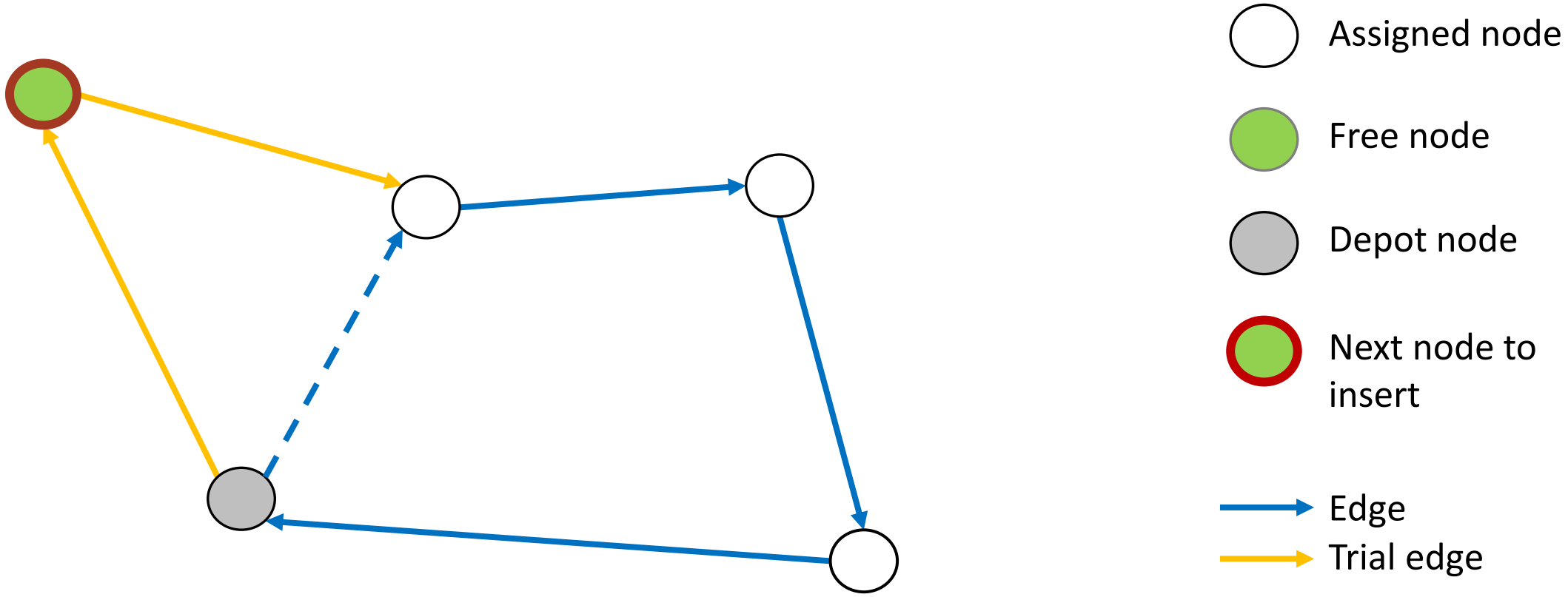
Insertion example 1 (6/12)



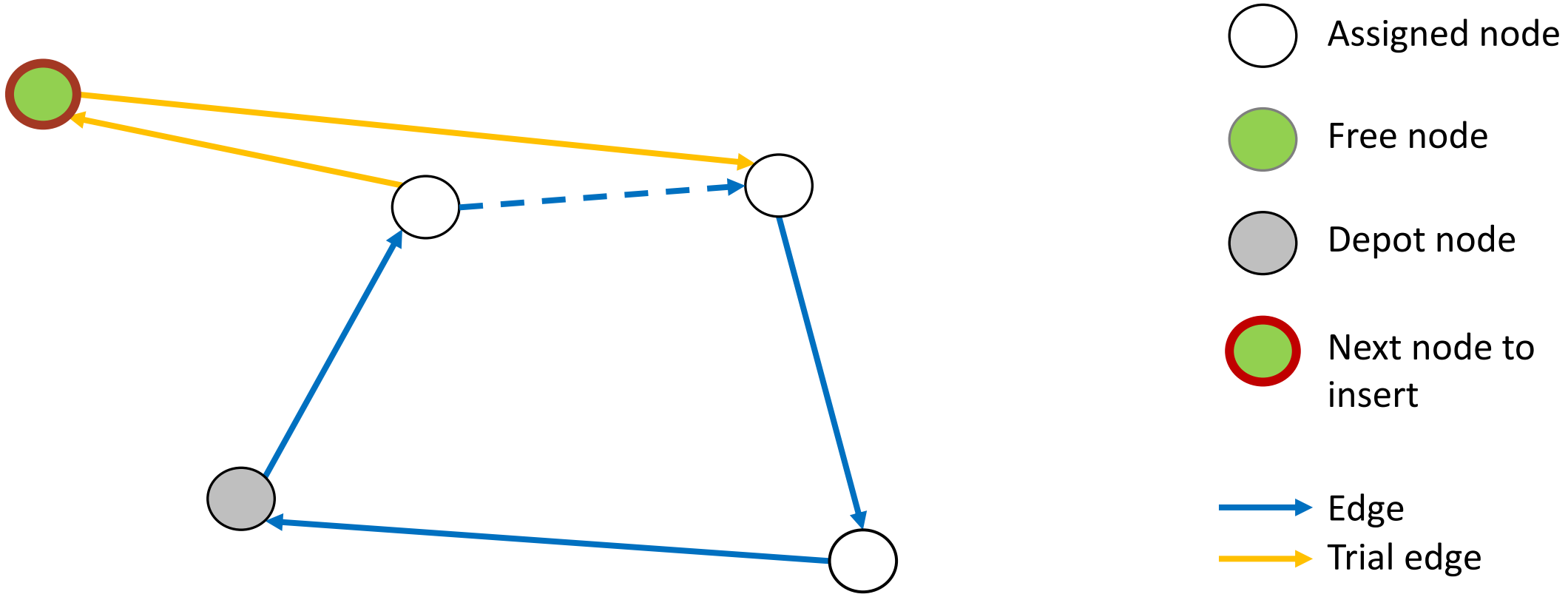
Insertion example 1 (7/12)



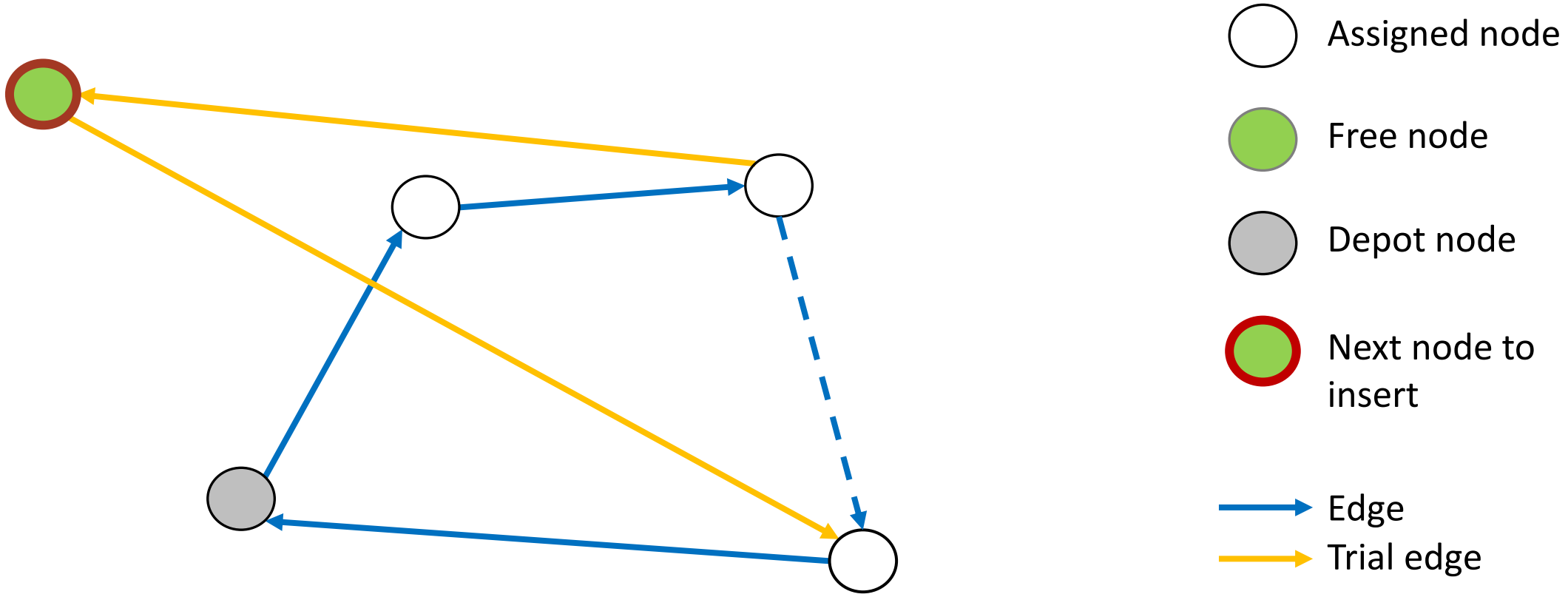
Insertion example 1 (8/12)



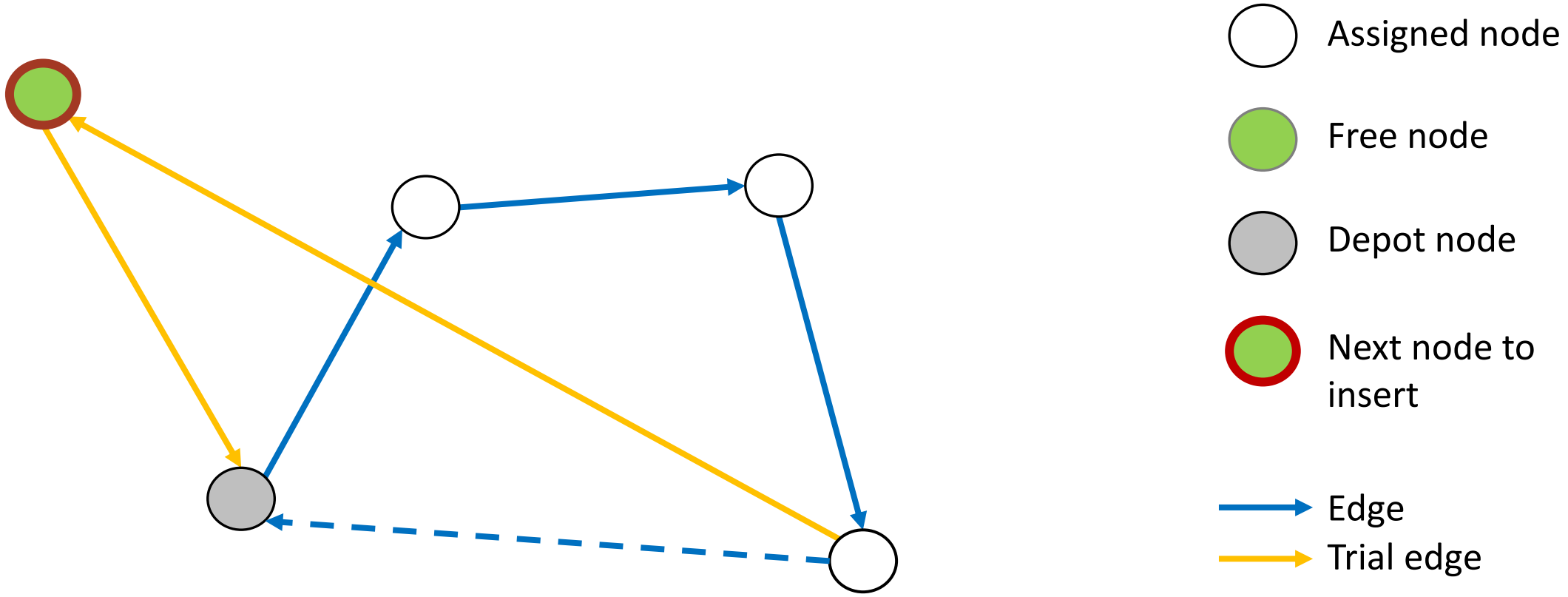
Insertion example 1 (9/12)



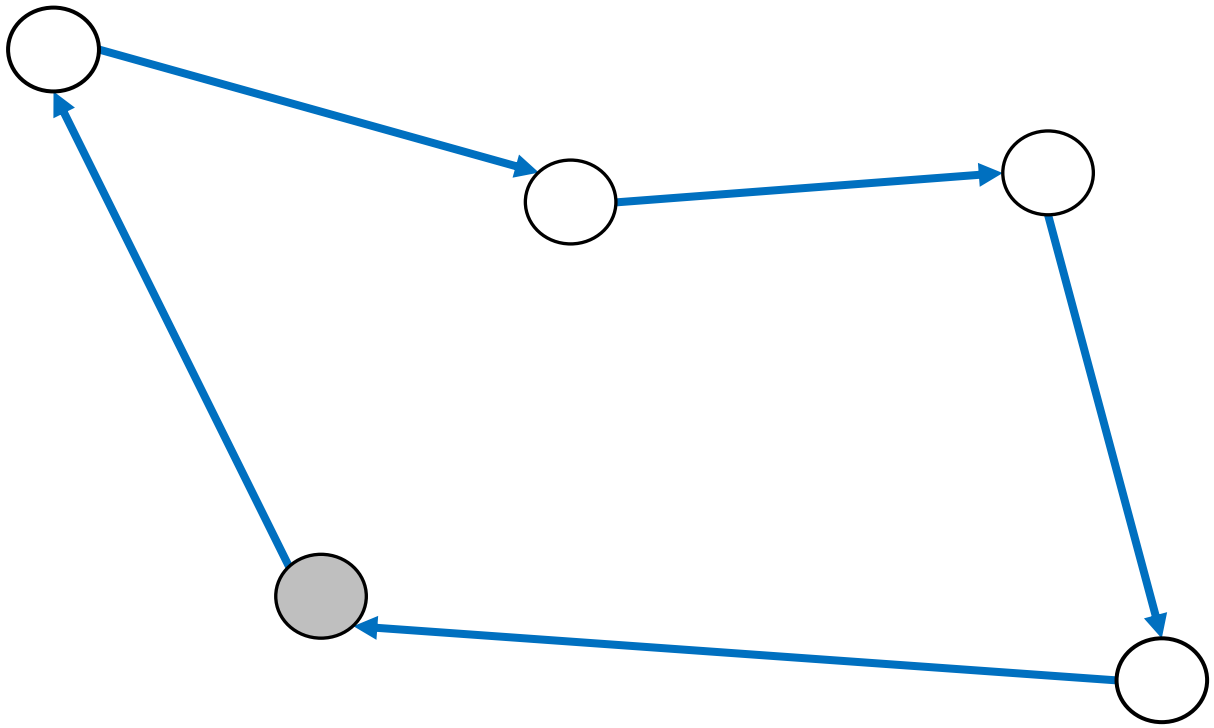
Insertion example 1 (10/12)

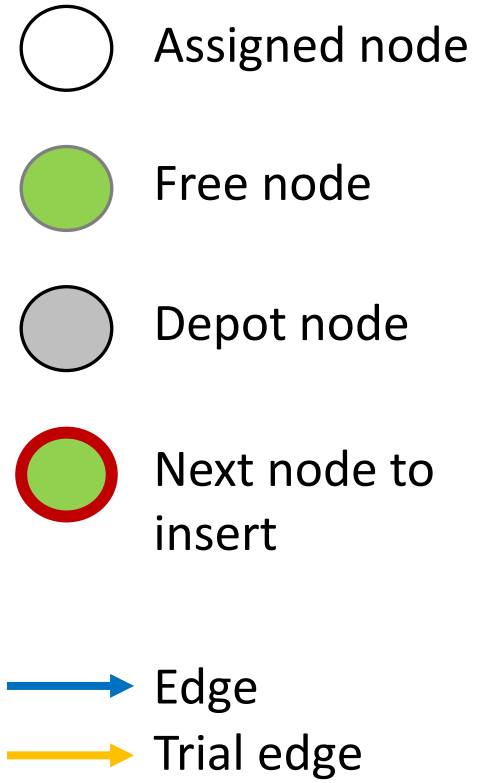


Insertion example 1 (11/12)

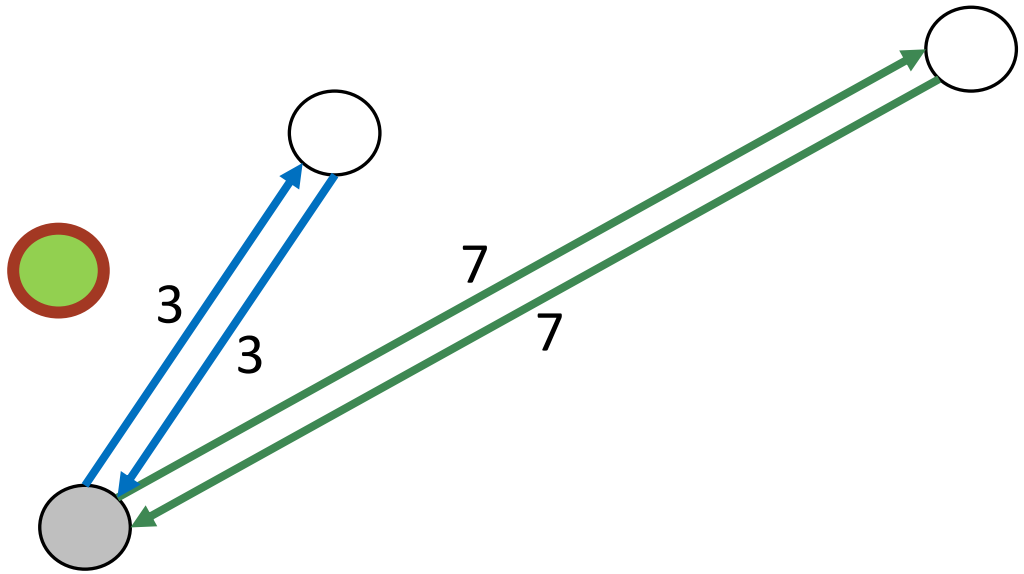


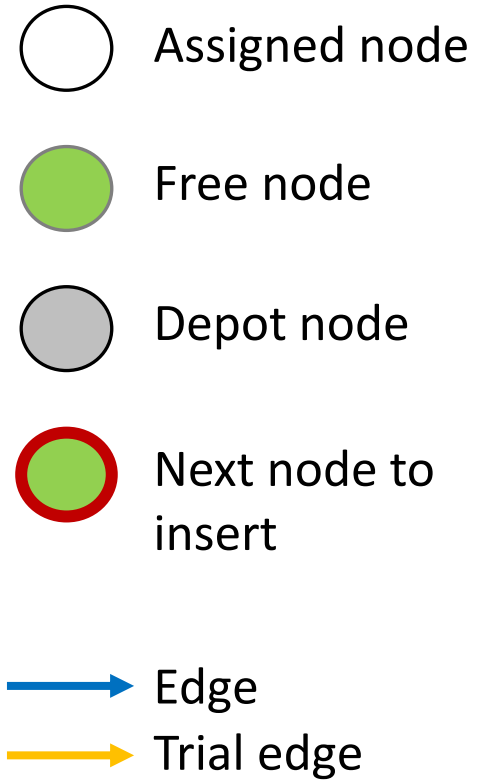
Insertion example 1 (12/12)



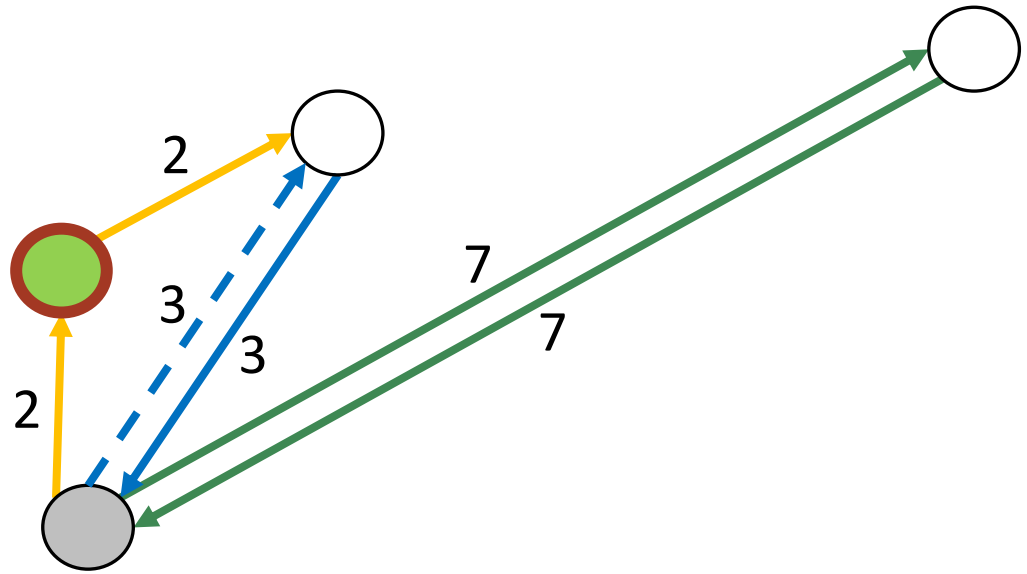


Insertion example 2 (1/4)





Insertion example 2 (2/4)



Assigned node

Free node

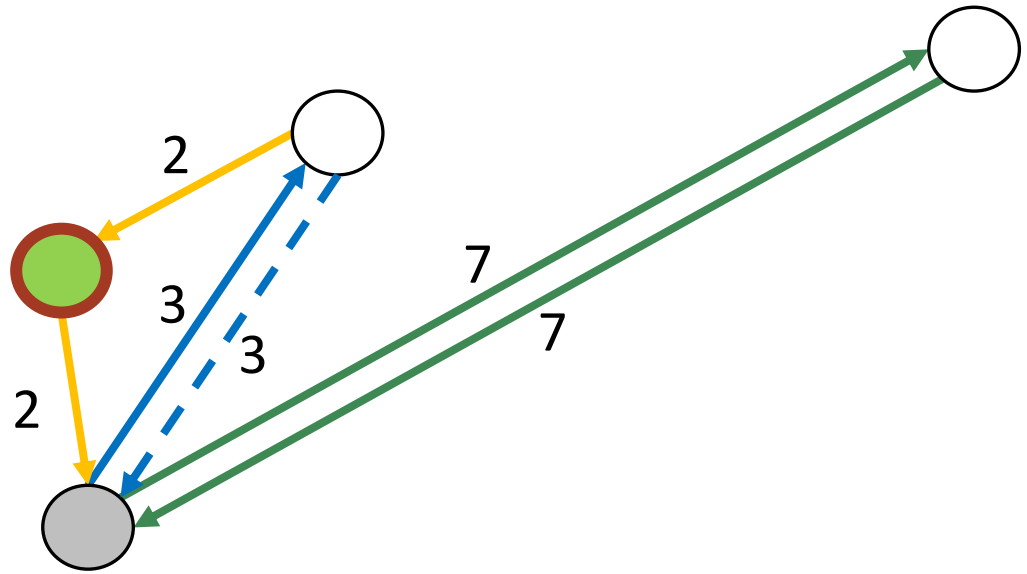
Depot node

Next node to insert

Edge

Trial edge

Insertion example 2 (3/4)



Assigned node

Free node

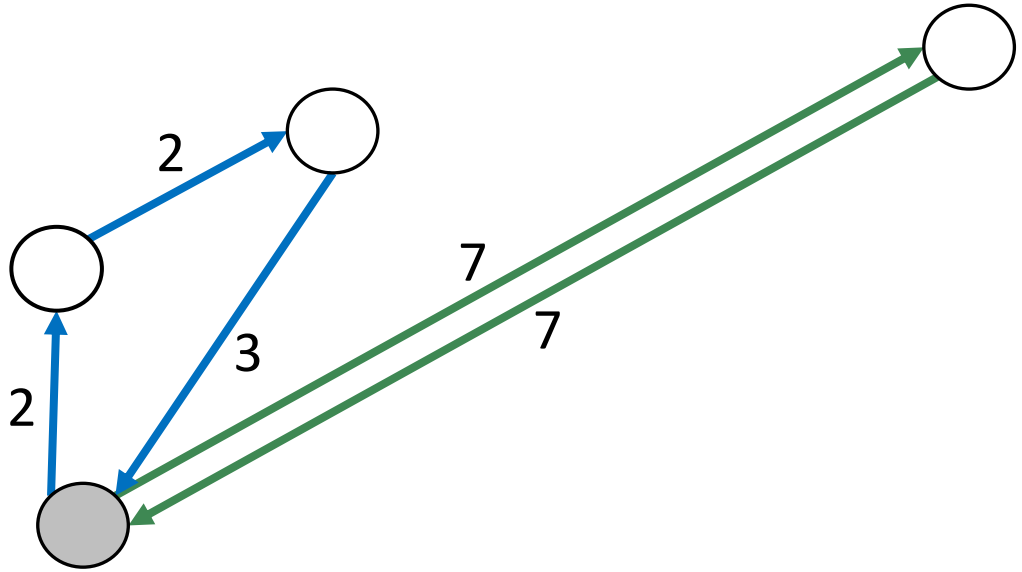
Depot node

Next node to insert

Edge

Trial edge

Insertion example 2 (4/4)



Assigned node

Free node

Depot node

Next node to insert

Edge

Trial edge

- Motivation
- Problem formulation
- Approach
- Experiments and results
- Conclusion

Experimental setup

- 4 Euclidean benchmarks & 3 general benchmarks
  - eil51, kroB100, ch150, lin318
  - kro124p, gr120, ftv170

- Reference: Invasive Weed Optimization (IWO)

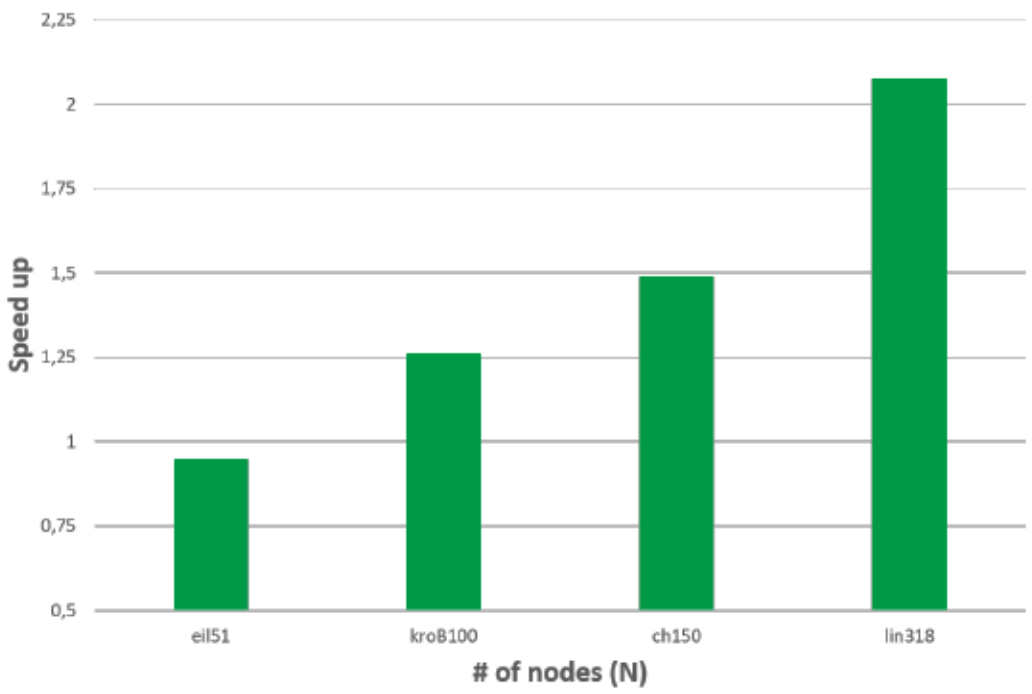
- Configuration of TS-LNS
  - *PopSize* = 100, *MinPop* = 6, *f* = 2
  - *mu*t = 200 on 1st iteration and increase by 200 on each iteration

Results: Euclidean benchmarks

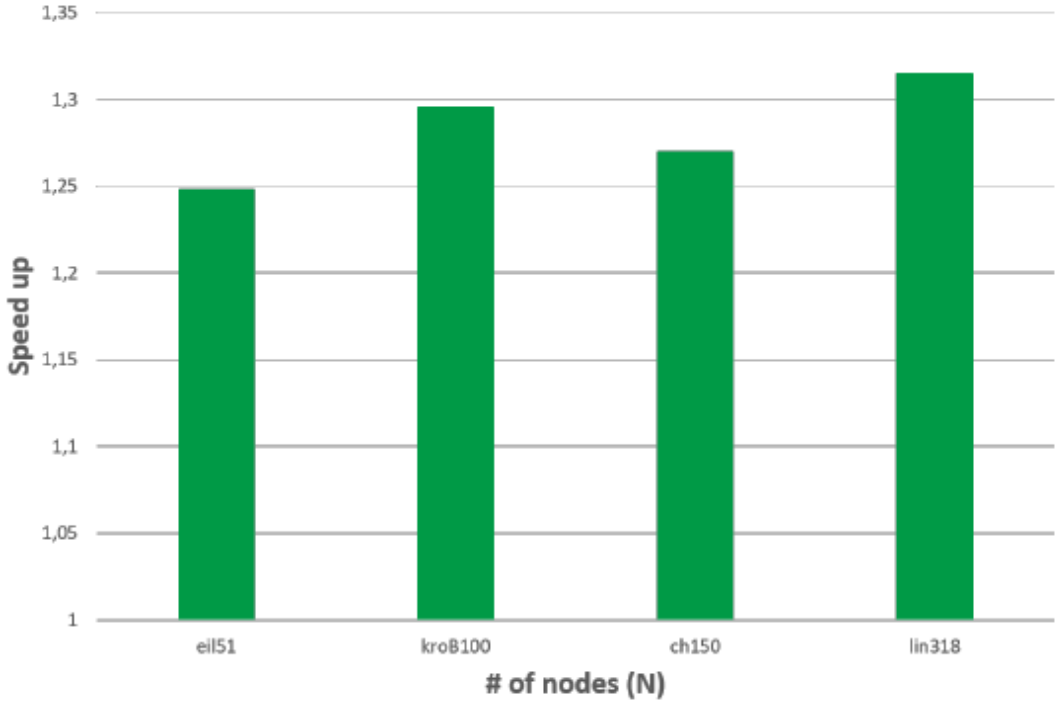
Benchmark	M	TS-LNS Euclidean			TS-LNS general			IWO		
		Cost Avg	Cost StD	Time (s)	Cost Avg	Cost StD	Time (s)	Cost Avg	Cost StD	Time (s)
eil51	3	159.56	0	13.98	159.56	0	13.39	159.56	0	16.58
	5	118.13	0	15.87	118.13	0	15.03	118.13	0	18.71
	10	112.08	0	19.41	112.08	0	18.13	112.08	0	22.86
kroB100	3	8482.50	5.88	34.43	8497.79	19.69	43.98	8503.41	22.73	56.69
	5	6965.85	17.56	38.81	6982.58	17.05	48.02	7008.01	15.06	63.61
	10	6700.04	0	46.98	6700.04	0	59.40	6700.04	0	75.57
ch150	3	2416.55	13.47	65.09	2446.41	15.03	97.41	2455.21	20.66	124.20
	5	1747.36	5.66	72.32	1764.80	8.68	107.05	1768.66	8.86	135.26
	10	1554.64	0	86.62	1554.64	0	128.31	1554.64	0	162.98
lin318	3	16113.78	46.12	215.62	16556.07	109.40	451.50	17138.27	161.83	593.22
	5	11500.98	43.78	236.42	11701.93	53.67	486.72	12379.09	68.36	651.16
	10	9731.16	0	281.31	9731.16	0	581.72	9816.99	20.62	751.88

Speedup: Euclidean benchmarks

TS-LNS Euclidean vs TS-LNS general



TS-LNS general vs IWO



Results: general benchmarks

Benchmark	M	IWO	TS-LNS general	TS-LNS Euclidean
		Cost Average		
kro124p	3	13470.05	13539.90	13313.2
	5	9137.3	9157.15	8990.55
	10	6419.4	6343.45	6322.45
gr120	3	2614.15	2604.95	2580.50
	5	1834.25	1823.0	1812.30
	10	1558.0	1554.40	1555.35
ftv170	3	1026.75	1026.1	986.65
	5	688.85	673.63	654.15
	10	447.70	432.37	427.53

- Motivation
- Problem formulation
- Approach
- Experiments and results
- Conclusion

Conclusion

- New hybrid algorithm
- Combining Tournament Selection and Large Neighborhood Search
- Produces solutions of better or equal quality compared to IWO
- Much faster than IWO

- Future work
  - Test more sophisticated removal/insertion functions
  - Extend to tackle problems with multiple depots / capacitated vehicles

Acknowledgements

This research has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE, project PV-Auto-Scout, code T1EDK-02435.