Ben-Gurion University of the Negev

CP 2024 GIRONA

Latency-Aware 2-Opt Monotonic Local Search for Distributed Constraint Optimization

Ben Rachmut

Joint work with:

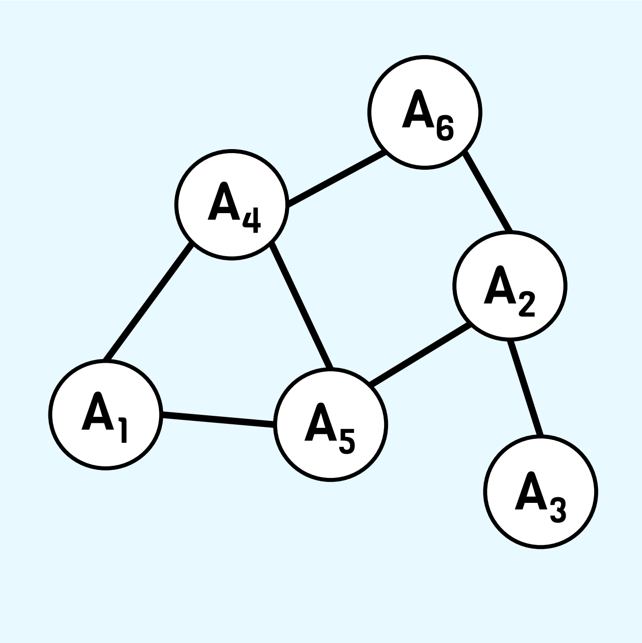
Roie Zivan and William Yeoh

Ben-Gurion University of the Negev

Washington University in St.Louis

Overview

Distributed Multi-Agent Systems:



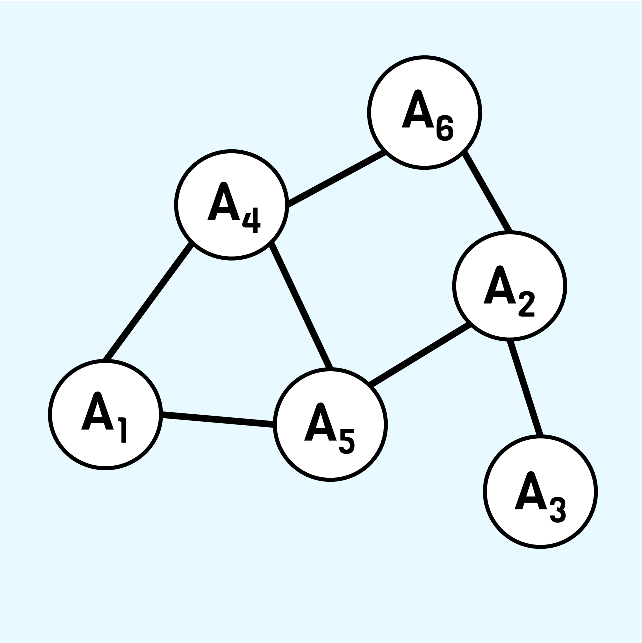
IoT applications

Agents cooperate and communicate

Mutual goal

Overview

Distributed Multi-Agent Systems:



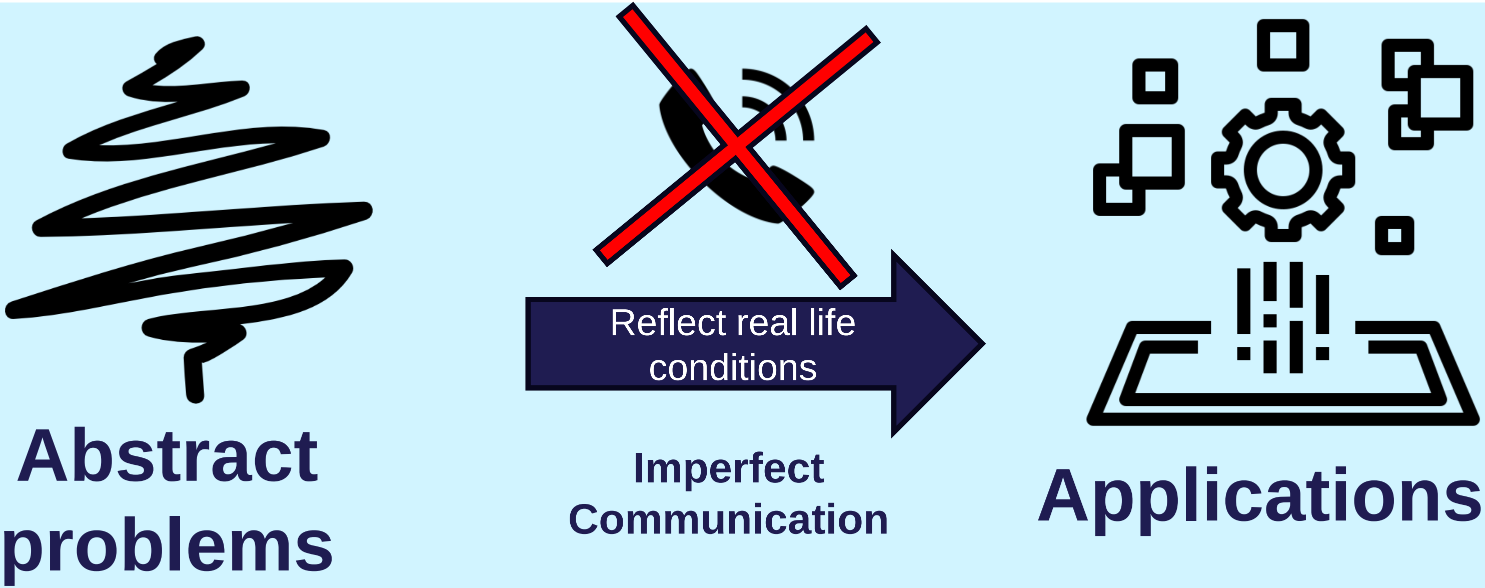
IoT applications

Agents cooperate and communicate

Mutual goal

DCOP

Overview



Abstract problems

Reflect real life conditions

Imperfect Communication

Applications

Communication Assumptions

Unrealistic Assumptions: All messages arrive instantaneously.

Existing local search algorithms leverage this assumption.

Example: MGM ensures monotonicity and convergence to a 1-opt solution

Communication Assumptions

Unrealistic Assumptions: All messages arrive instantaneously.

Existing local search algorithms leverage this assumption.

Example: MGM ensures monotonicity and convergence to a 1-opt solution

Communication Assumptions

Unrealistic Assumptions: All messages arrive instantaneously.

Existing local search algorithms leverage this assumption.

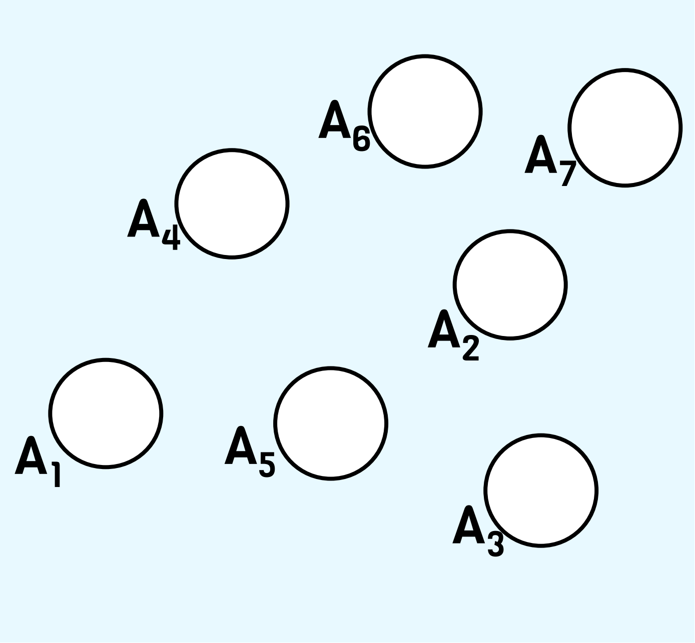
Example: MGM ensures monotonicity and convergence to a 1-opt solution

Background

Distributed Constraint Optimization Problem (DCOP)

DCOP is a tuple:

A - Agents $\{A_1,\ldots, A_n\}$

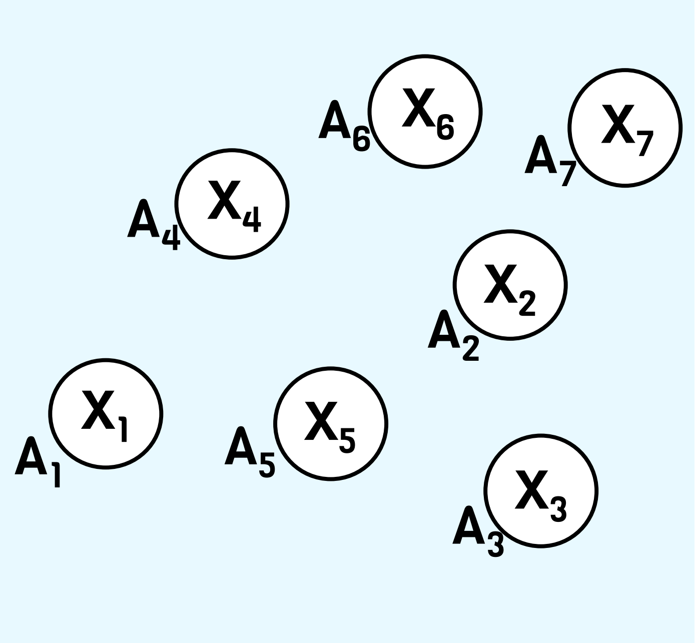


Distributed Constraint Optimization Problem (DCOP)

DCOP is a tuple:

A - Agents $\{A_1,\ldots, A_n\}$

X - Variables $\{X_1,\ldots, X_m\}$



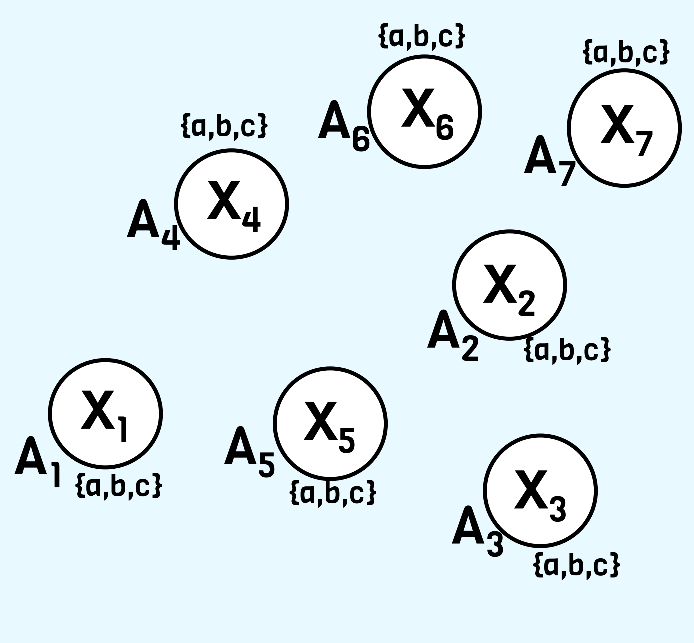
Distributed Constraint Optimization Problem (DCOP)

DCOP is a tuple:

A - Agents $\{A_1,\ldots, A_n\}$

X - Variables $\{X_1,\ldots, X_m\}$

D - Domain $\{D_1,\ldots, D_m\}$



Distributed Constraint Optimization Problem (DCOP)

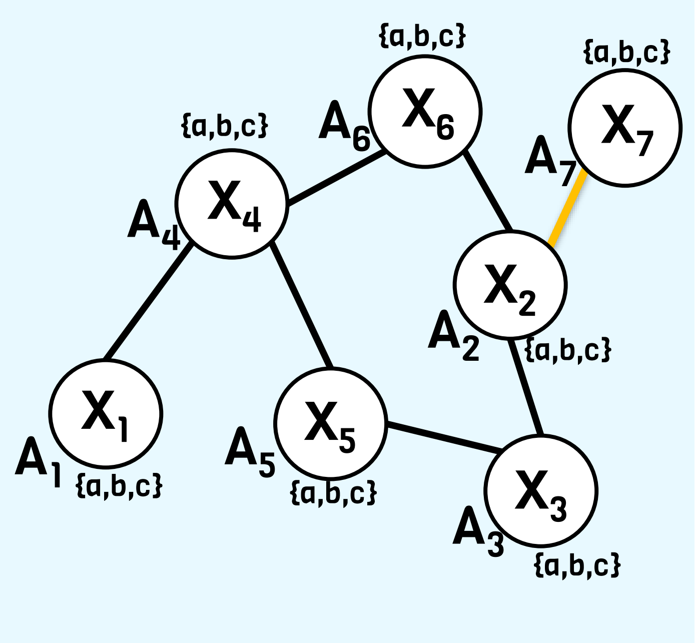
DCOP is a tuple:

A - Agents $\{A_1,\ldots, A_n\}$

X - Variables $\{X_1,\ldots, X_m\}$

D - Domain $\{D_1,\ldots, D_m\}$

C- Constraints



C_{27}

X_2/ X_7	a	b	c
a	8	3	2
b	1	5	0
c	6	0	4

Distributed Constraint Optimization Problem (DCOP)

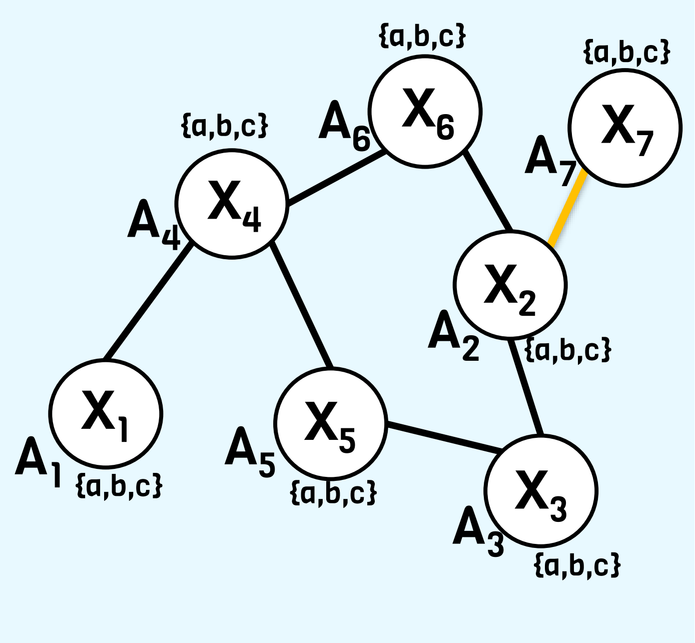
DCOP is a tuple:

A - Agents $\{A_1,\ldots, A_n\}$

X - Variables $\{X_1,\ldots, X_m\}$

D - Domain $\{D_1,\ldots, D_m\}$

C- Constraints

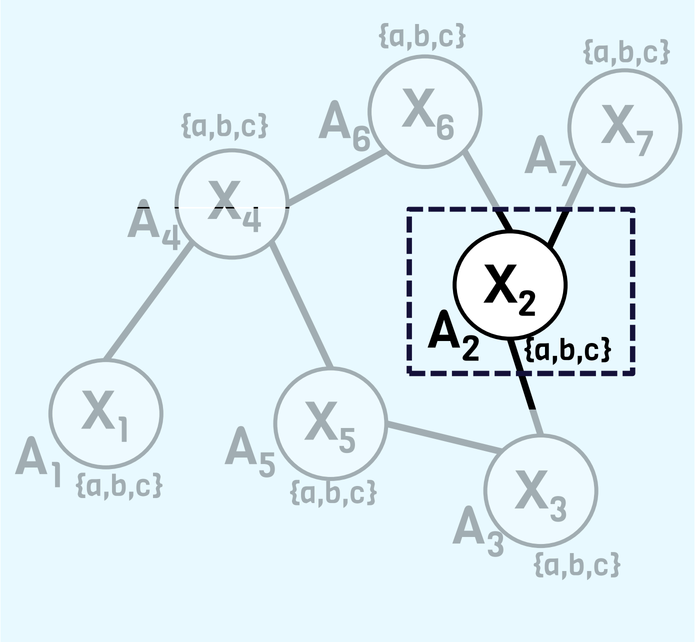


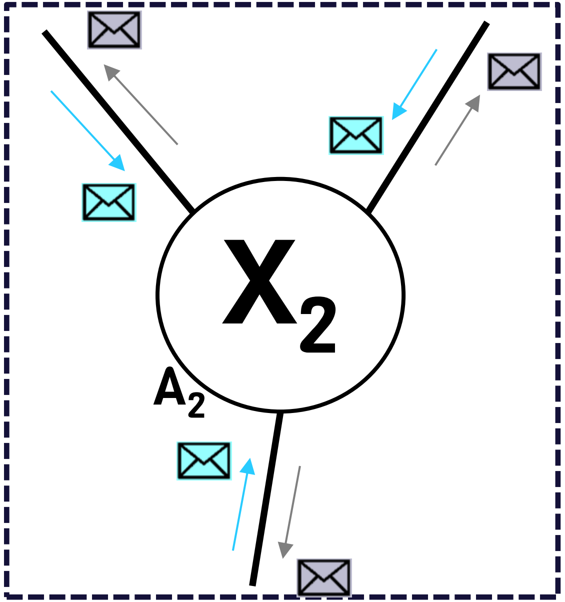
C_{27}

X_2/ X_7	a	b	c
a	8	3	2
b	1	5	0
c	6	0	4

Goal:
finding a complete assignment with minimal global cost

Solving DCOPs

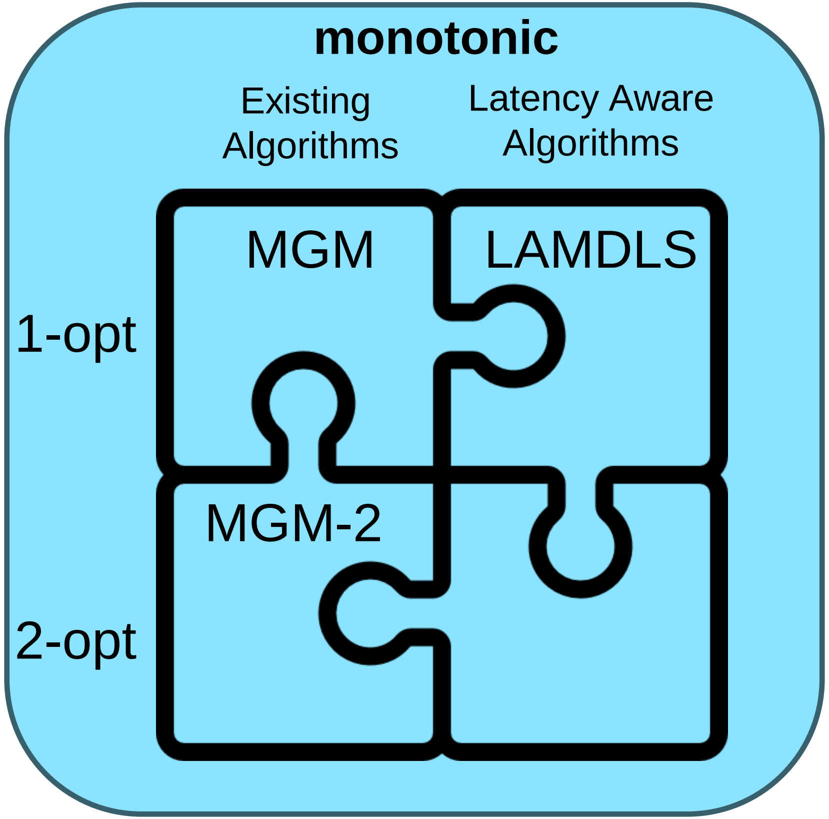




Solving DCOPs

Desirable properties

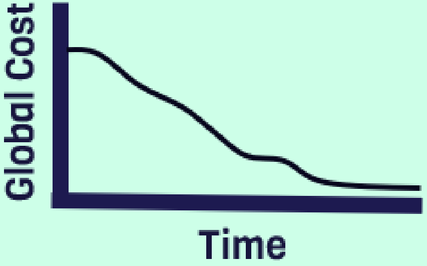
- Monotonicity
- Convergence to a 1 and 2 opt solutions

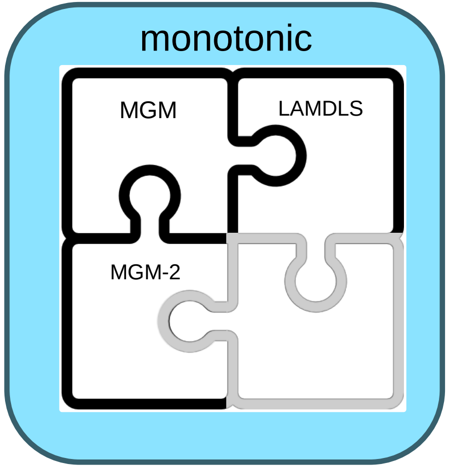


Solving DCOPs

Weakly Monotonic

When an agent changes an assignment the global cost decreases or stays the same.

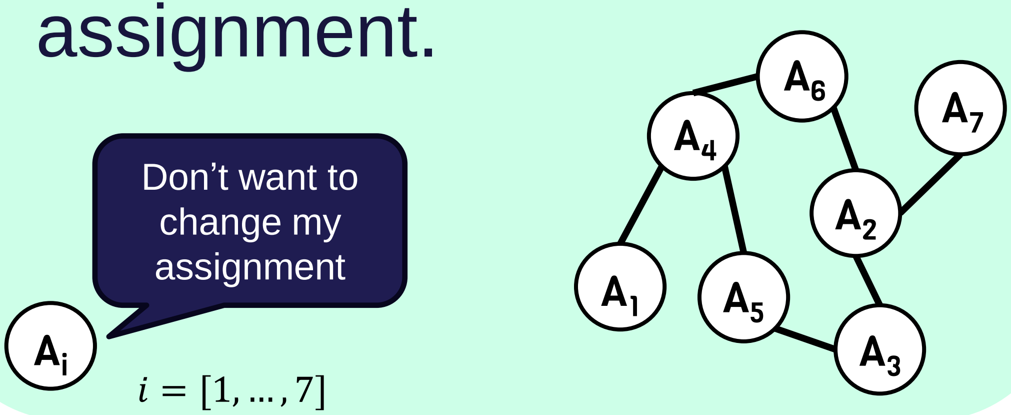


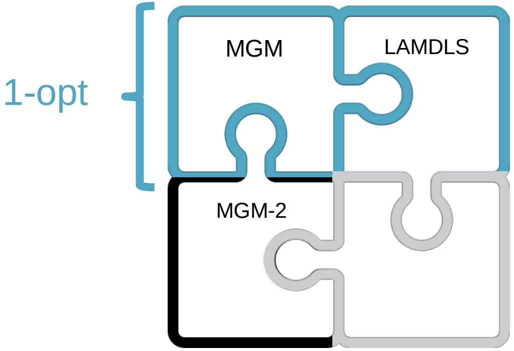


Solving DCOPs

1 opt convergence

No single agent can improve the solution by changing its assignment.

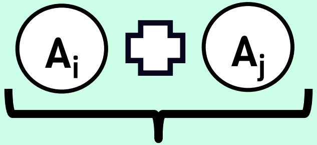


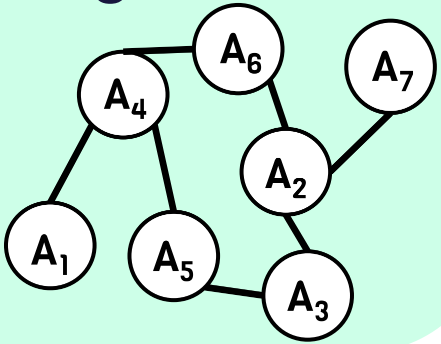


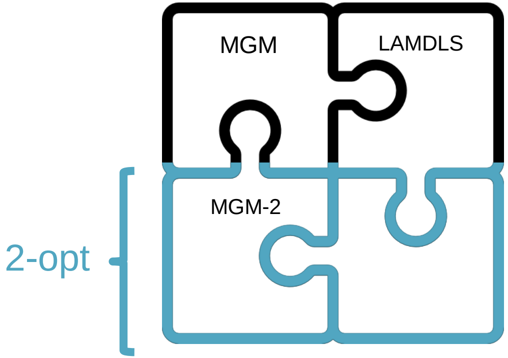
Solving DCOPs

2 opt convergence

No subset of 2 agents can improve the solution by changing their assignments.







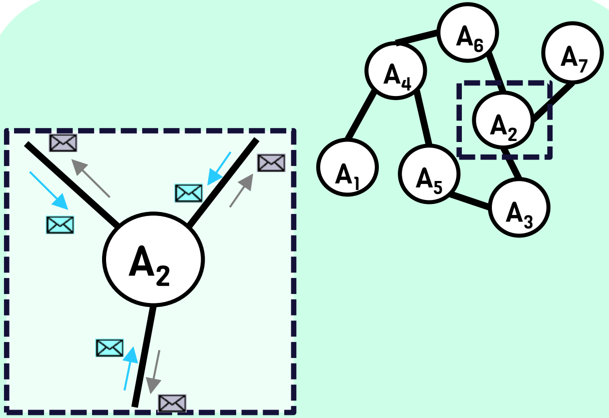
Challenge: form **all** possible pairs

MGM

Exchange messages about potential local reductions with their neighbors.

The agent with the maximum reduction changes its value assignment.

A single agent in a neighborhood changes its value, ensuring monotonicity.



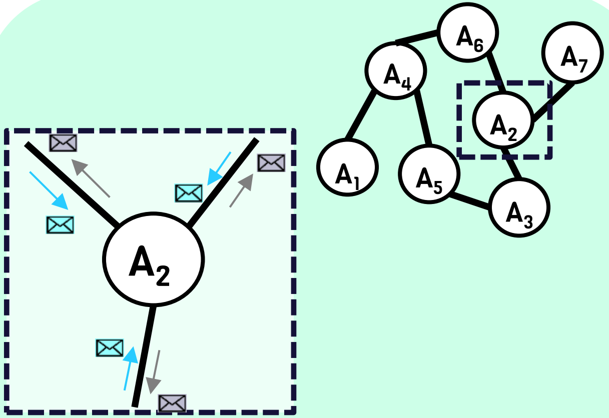
$LR_2 = 10\colon [LR_7 = 2, LR_6 = 4, LR_3 = 9]$

MGM

Exchange messages about potential local reductions with their neighbors.

The agent with the maximum reduction changes its value assignment.

A single agent in a neighborhood changes its value, ensuring monotonicity.



$$LR_2 = 10\!: [LR_7 = 2, LR_6 = 4, LR_3 = 9]$$

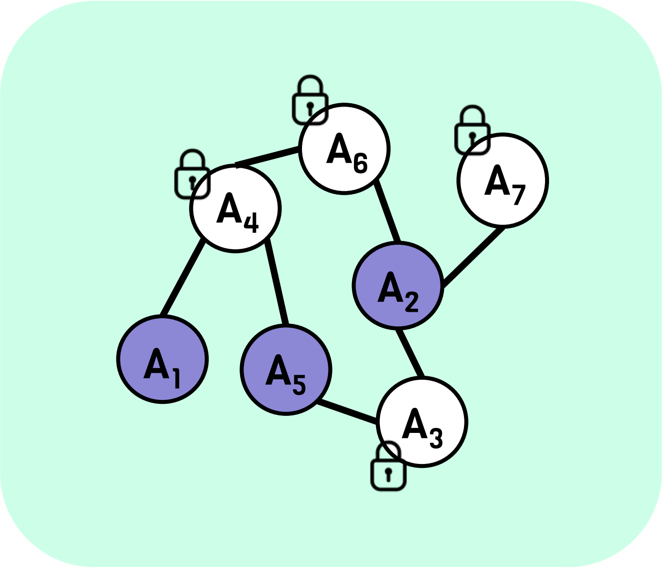
$$LR_2 > LR_7, LR_6, LR_3$$

MGM

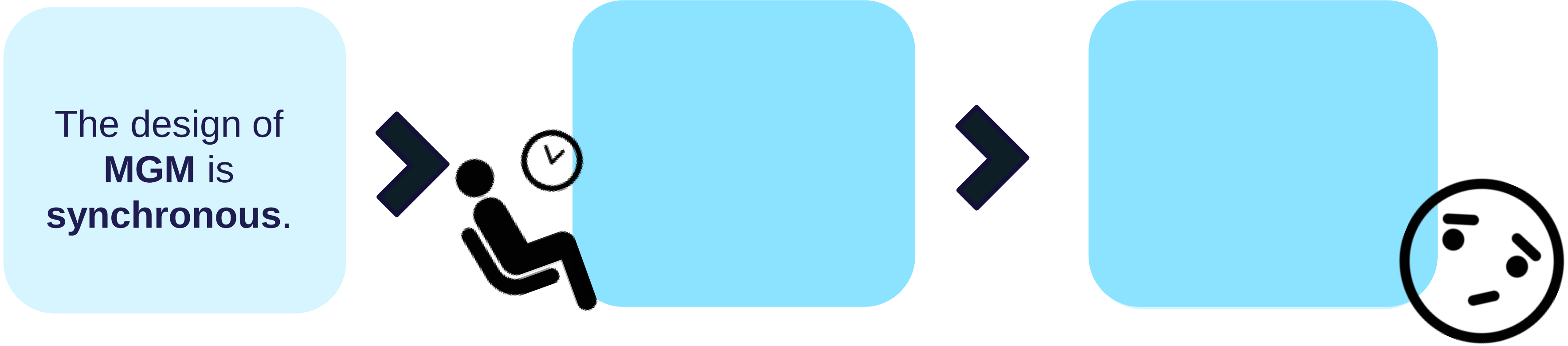
Exchange messages about potential local reductions with their neighbors.

The agent with the maximum reduction changes its value assignment.

A single agent in a neighborhood changes its value, ensuring monotonicity.

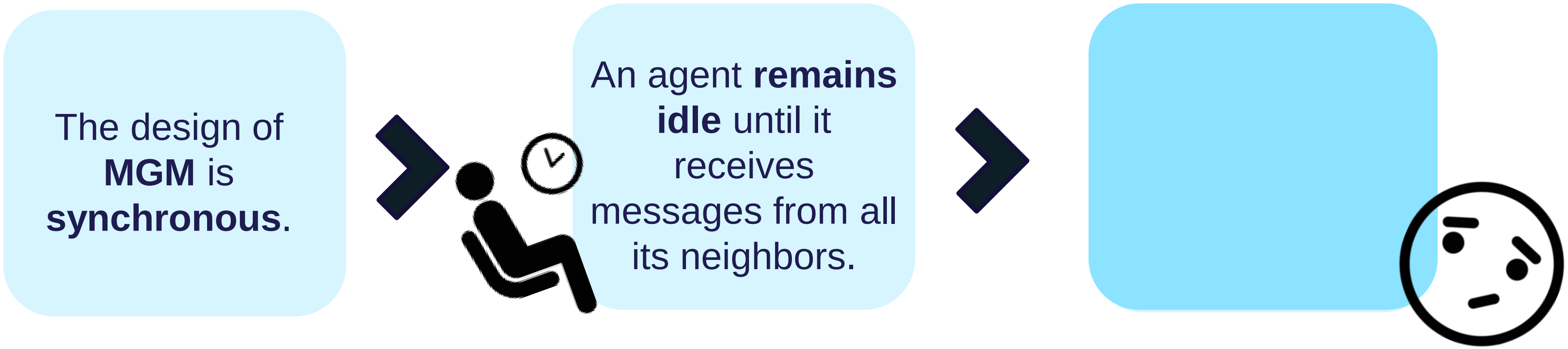


MGM – Synchronous design



The design of **MGM** is **synchronous.**

MGM – Synchronous design



The design of **MGM** is **synchronous**.

An agent **remains idle** until it receives messages from all its neighbors.

MGM – Synchronous design

The design of **MGM** is **synchronous**.

An agent **remains idle** until it receives messages from all its neighbors.

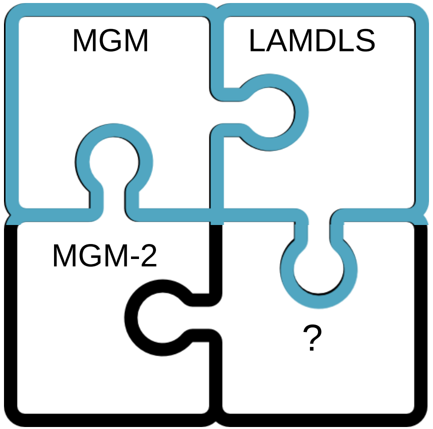
The algorithm advances based on the **longest** message **delay**

MGM – Synchronous design

Requires 2 synchronous iterations per cycle:

- 1 for local reduction
- 1 for value assignment change

Latency Aware Design



Perfect Communication

Message Latency

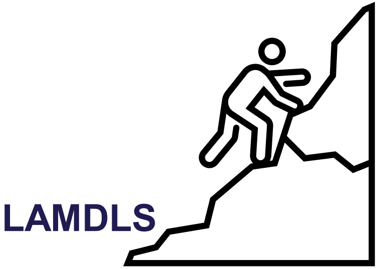
LAMDLS

Design an algorithm that guarantees the same properties and is aware of message latency.

Latency Aware Monotonic Distributed Local search

LAMDLS: monotonic and 1-opt

(Rachmut et al., JAIR 2022)

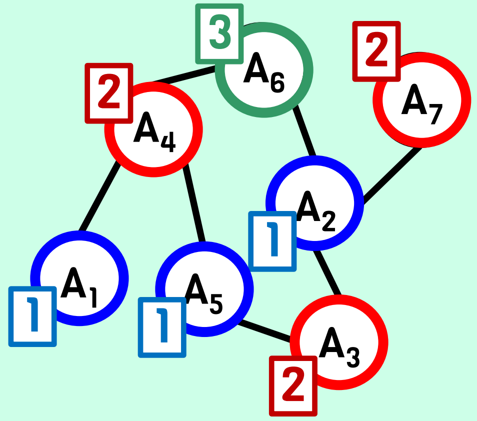


LAMDLS

Use Distributed Ordered Coloring Selection (DOCS) at the **beginning** of the algorithm to set an order.

Each agent divides neighbors into subsets using their color indexes.

Establish a defined order for decision-making.



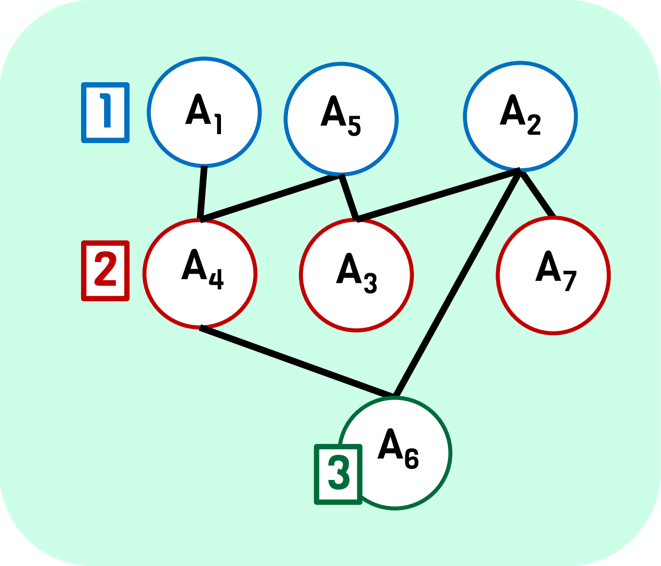
LAMDLS

LAMDLS

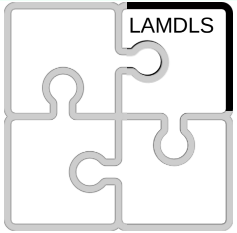
Use Distributed Ordered Coloring Selection (DOCS) at the **beginning** of the algorithm to set an order.

Each agent divides neighbors into subsets using their color indexes.

Establish a defined order for decision-making.



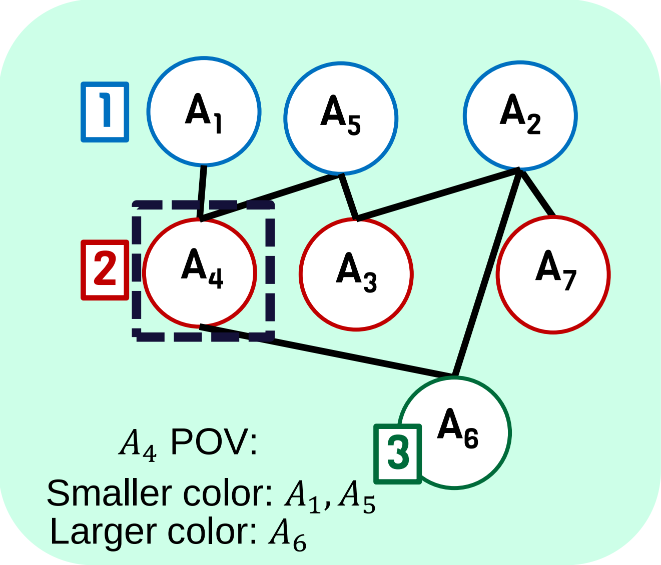
LAMDLS



Use Distributed Ordered Coloring Selection (DOCS) at the **beginning** of the algorithm to set an order.

Each agent divides neighbors into subsets using their color indexes.

Establish a defined order for decision-making.

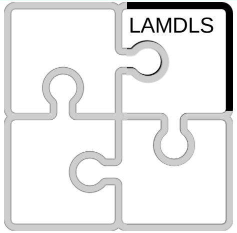


A_4 POV:

Smaller color: A_1, A_5

Larger color: A_6

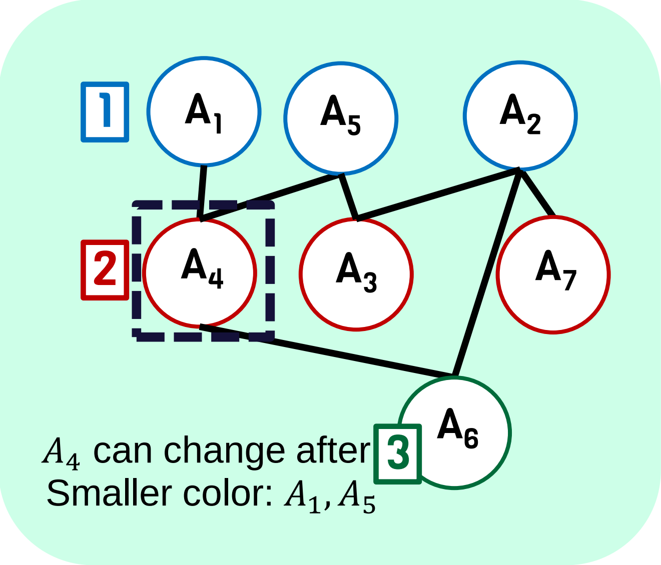
LAMDLS



Use Distributed Ordered Coloring Selection (DOCS) at the **beginning** of the algorithm to set an order.

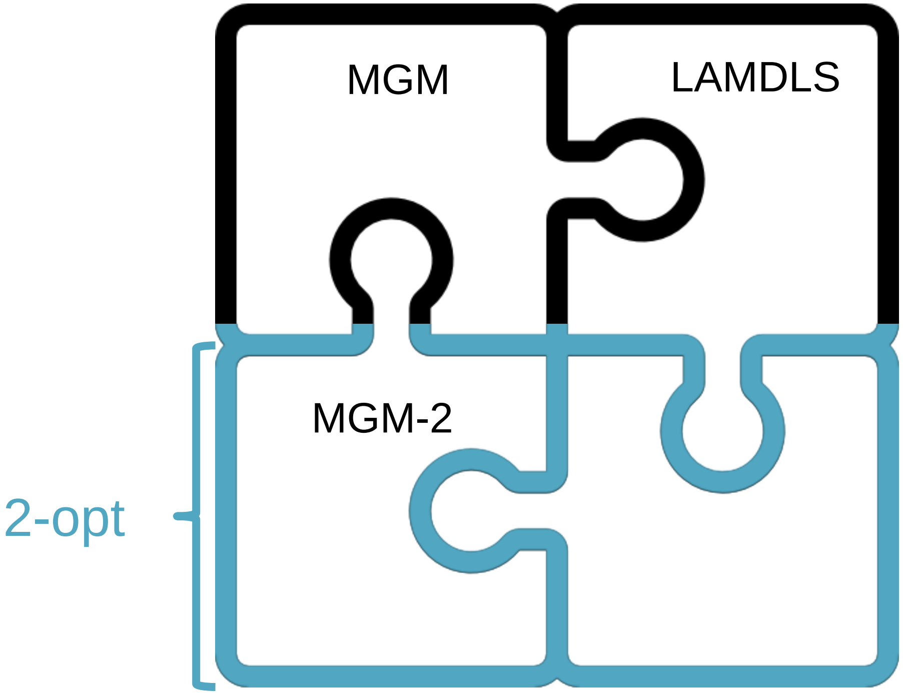
Each agent divides neighbors into subsets using their color indexes.

Establish a defined order for decision-making.



A_4 can change after
Smaller color: A_1, A_5

MGM-2

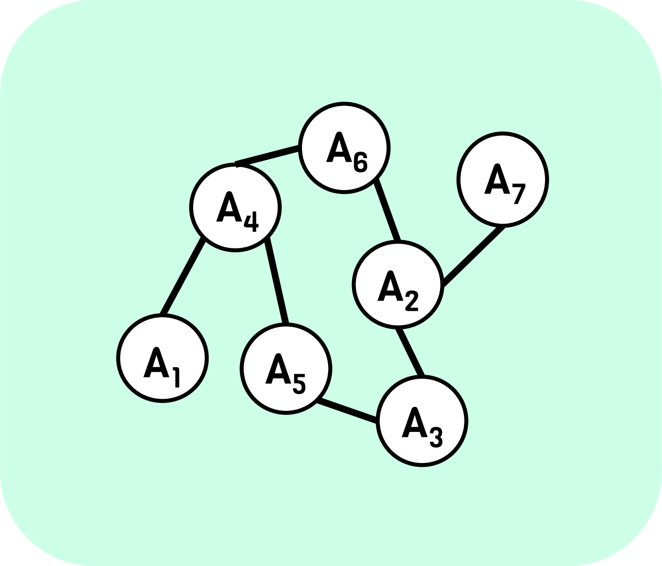


MGM-2

MGM 2

How can all possible pairs be identified?

- Agents pair up stochastically and exchange all information
- Find best Bilateral local reduction as a pair and exchange with neighbors
- Pair with the best local reduction changes values



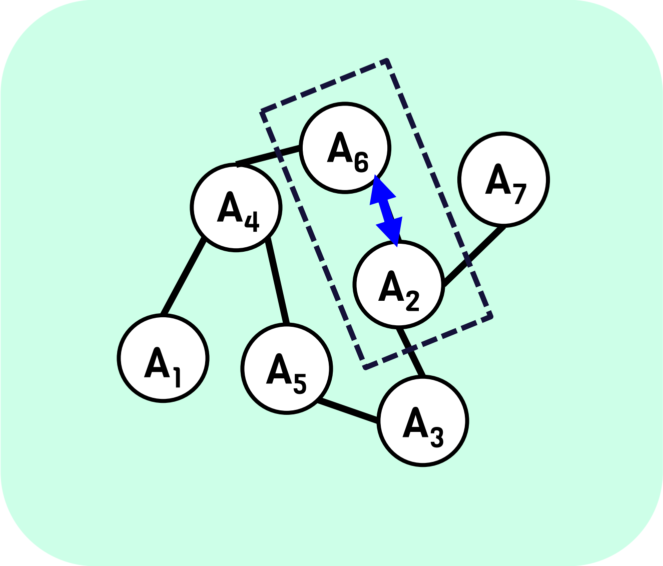
MGM-2

How can all possible pairs be identified?

Agents pair up stochastically and exchange all information

Find best Bilateral local reduction as a pair and exchange with neighbors

Pair with the best local reduction changes values



MGM-2

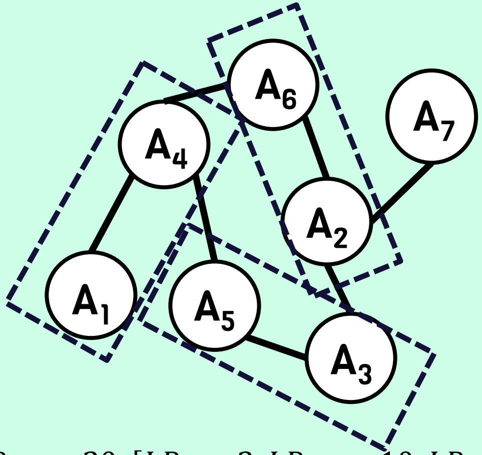
MGM 2

How can all possible pairs be identified?

Agents pair up stochastically and exchange all information

Find best Bilateral local reduction as a pair and exchange with neighbors

Pair with the best local reduction changes values



$LR_{2,6} = 20 : [LR_7 = 2, LR_{3,5} = 10, LR_{1,4} = 9]$

MGM-2

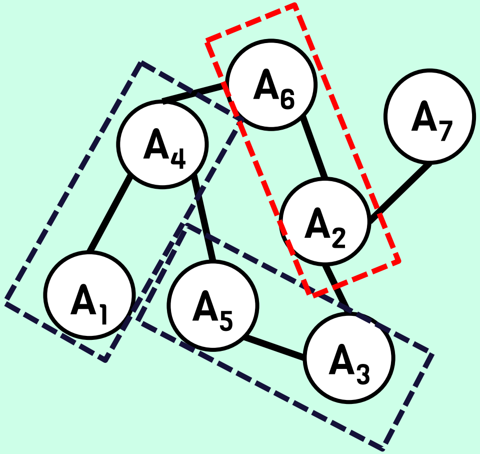
MGM
2

How can all possible pairs be identified?

Agents pair up stochastically and exchange all information

Find best Bilateral local reduction as a pair and exchange with neighbors

Pair with the best local reduction changes values

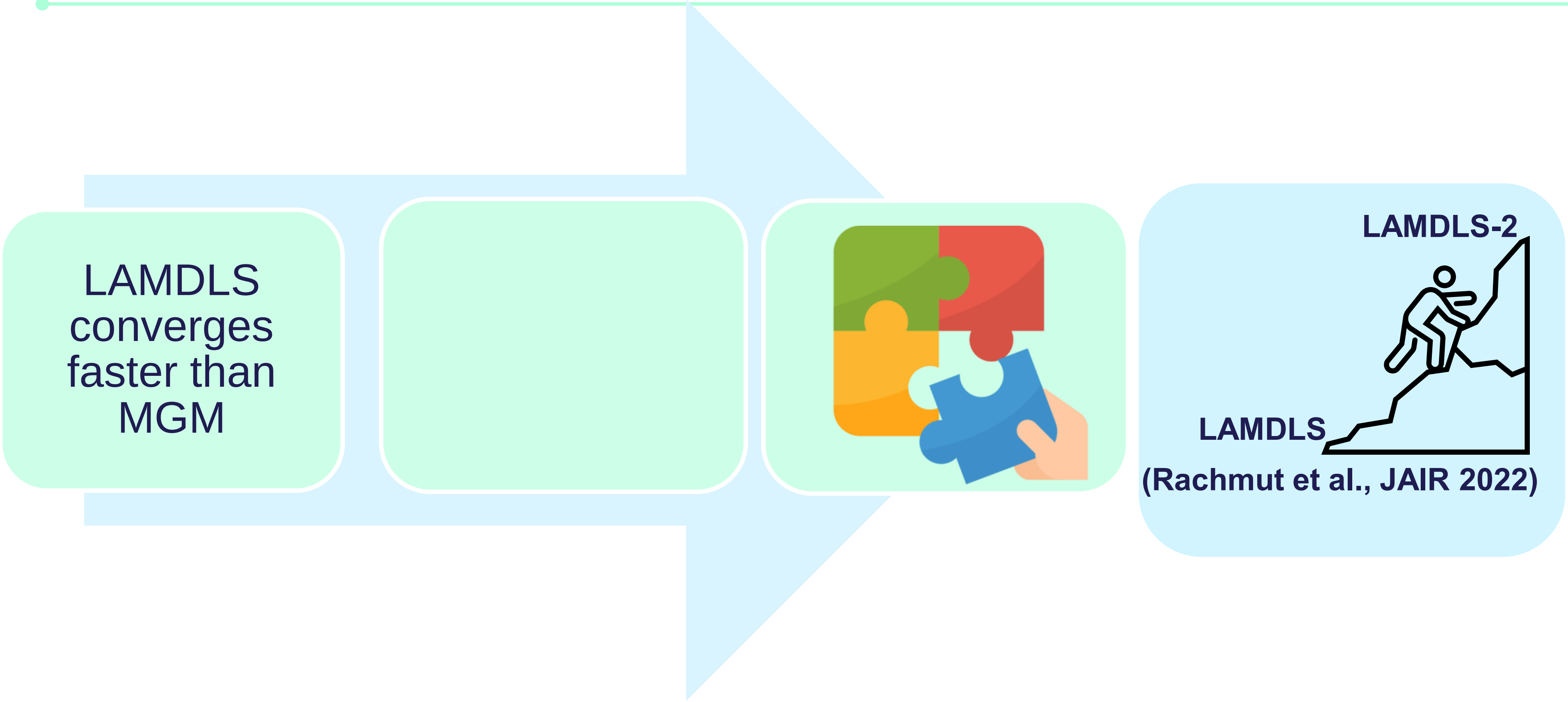


$LR_{2,6} = 20 > LR_7, LR_{3,5}, R_{1,4}$

MGM-2

Requires 5 synchronous iterations per cycle:
2 for coalition formation
3 for value exchange.

LAMDLS -> LAMDLS-2?



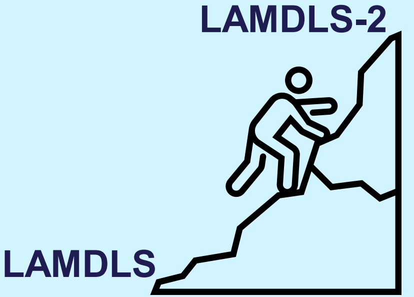
LAMDLS converges faster than MGM

(Rachmut et al., JAIR 2022)

LAMDLS -> LAMDLS-2?

LAMDLS converges faster than MGM

Guarantees monotonicity and convergence to a 1-opt



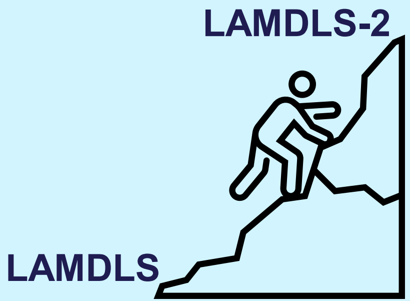
(Rachmut et al., JAIR 2022)

LAMDLS -> LAMDLS-2?

LAMDLS converges faster than MGM

Guarantees monotonicity and convergence to a 1-opt

Can extend it to 2-opt?



(Rachmut et al., JAIR 2022)

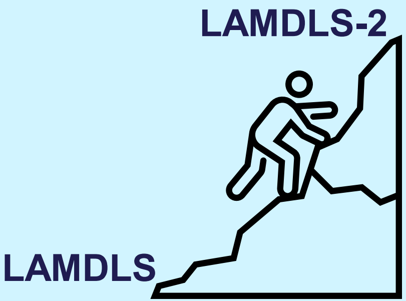
LAMDLS-2

LAMDLS-2

Challenges!

□ Form all possible combinations of coalitions.

□ Coordinate changes within a coalition; keep neighbors idle.



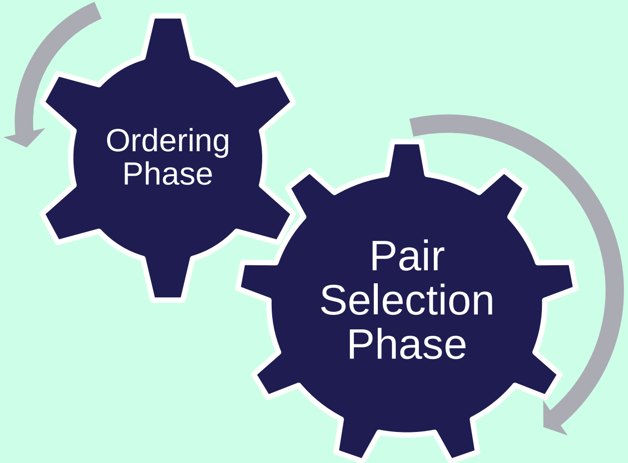
(Rachmut et al., JAIR 2022)

LAMDLS-2

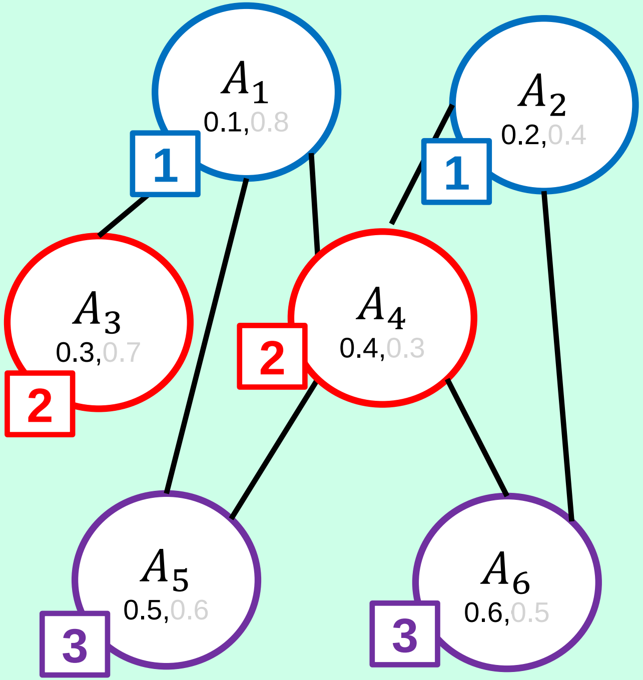
Challenges!

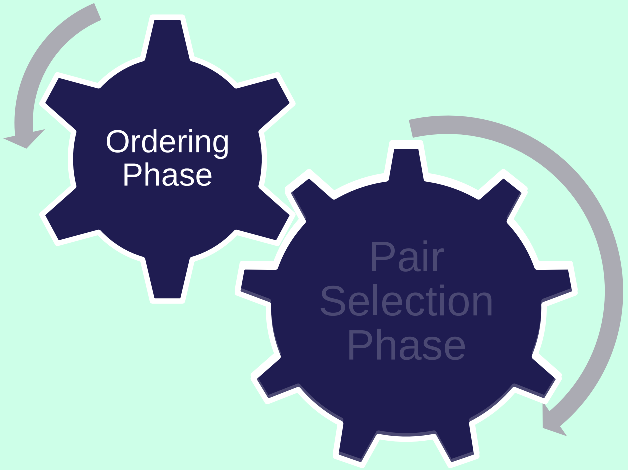
□ Form all possible combinations of coalitions.

□ Coordinate changes within a coalition; keep neighbors idle.

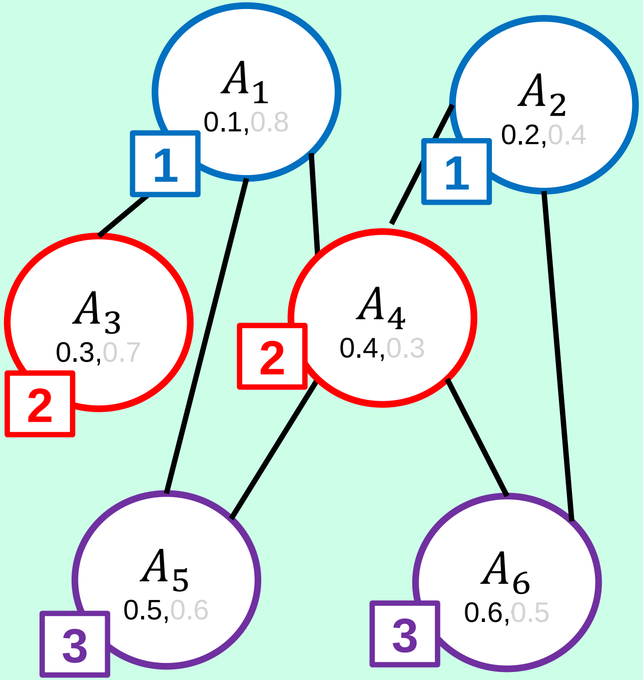


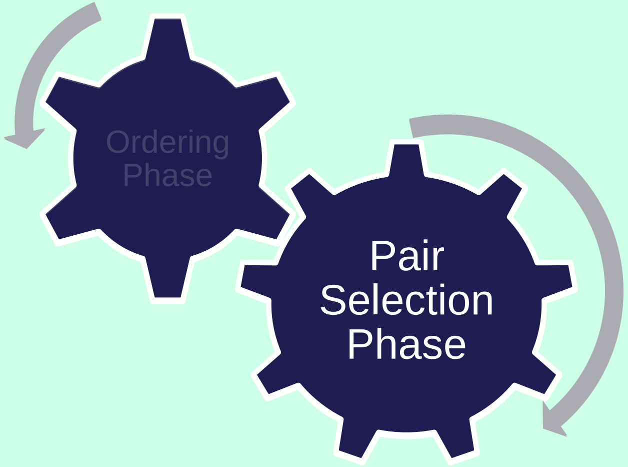
LAMDLS-2


A_1
0.1,0.8
1
A_2
0.2,0.4
1
A_3
0.3,0.7
2
A_4
0.4,0.3
2
A_5
0.5,0.6
3
A_6
0.6,0.5
3


Ordering Phase
Pair Selection Phase

LAMDLS-2

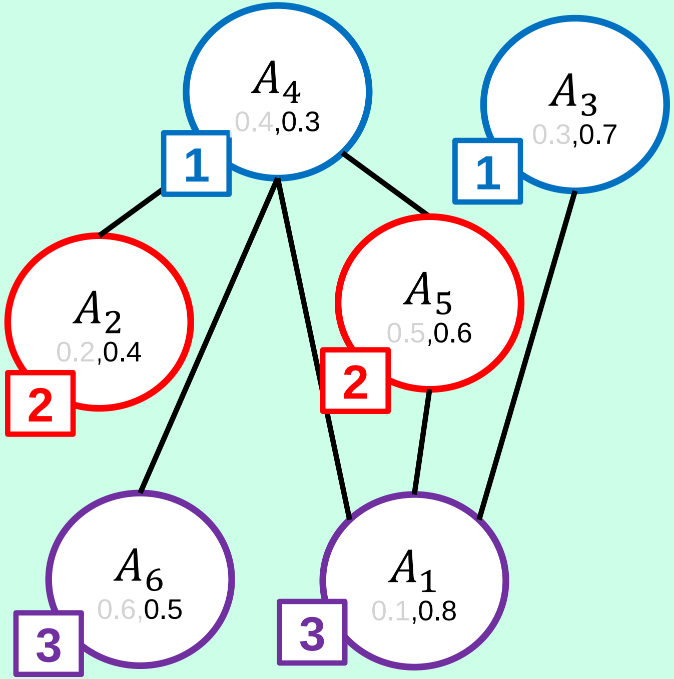


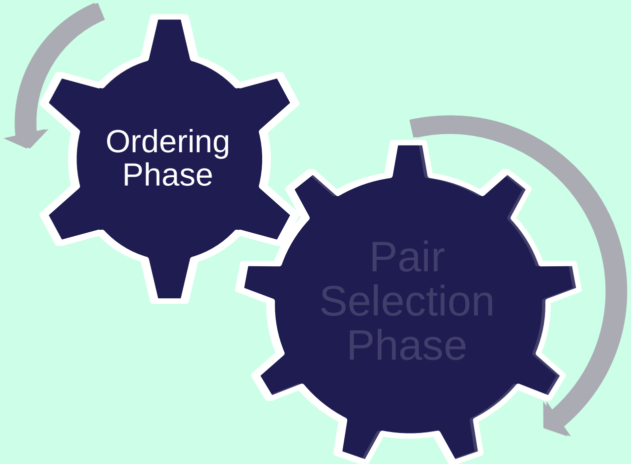


Ordering Phase

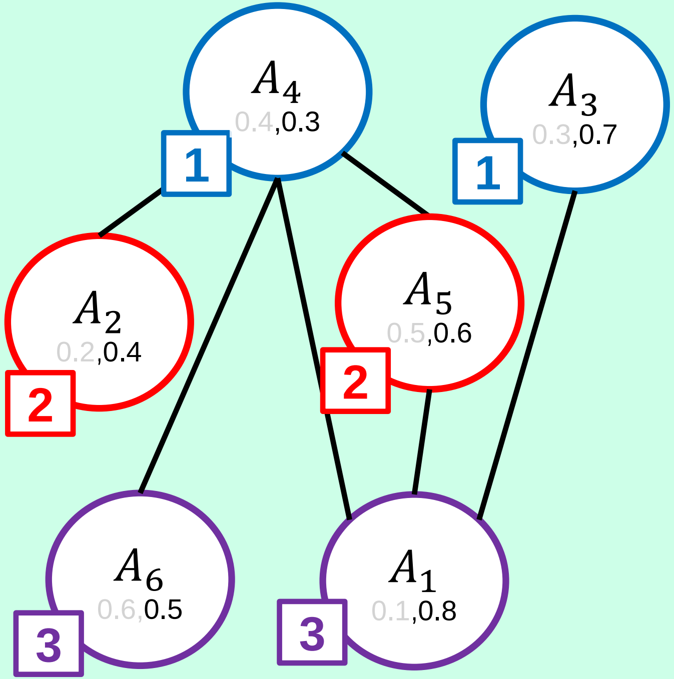
Pair Selection Phase

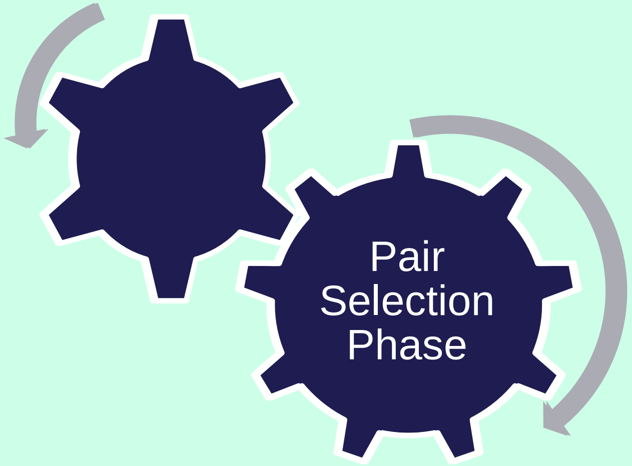
LAMDLS-2


A_4
0.4,0.3
1
A_3
0.3,0.7
1
A_2
0.2,0.4
2
A_5
0.5,0.6
2
A_6
0.6,0.5
3
A_1
0.1,0.8
3


Ordering Phase
Pair Selection Phase

LAMDLS-2


A_4
0.4,0.3
1
A_3
0.3,0.7
1
A_2
0.2,0.4
2
A_5
0.5,0.6
2
A_6
0.6,0.5
3
A_1
0.1,0.8
3


Pair Selection Phase

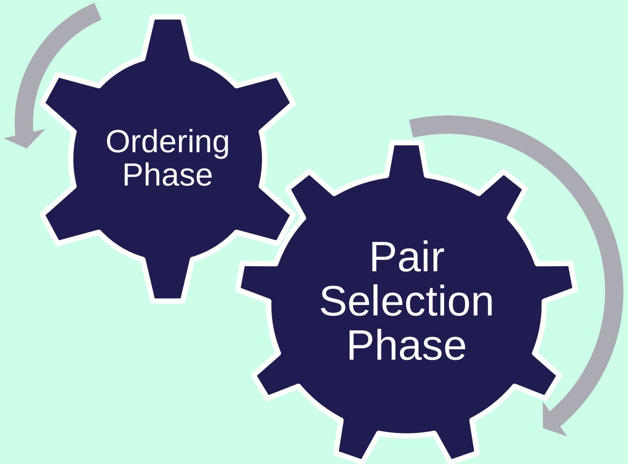
LAMDLS-2

Challenges!

□ ~~Form all possible combinations of coalitions.~~

“Manipulation on DOCS”

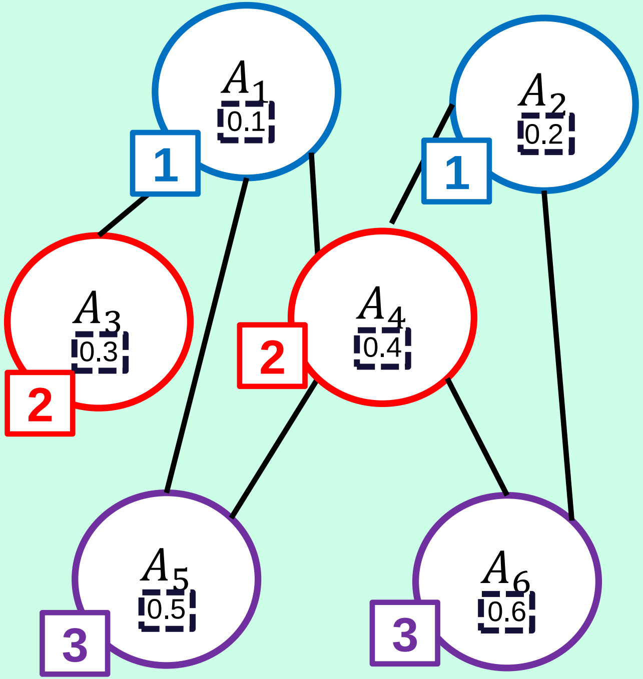
□ Coordinate changes within a coalition; keep neighbors idle.



LAMDLS-2

Challenges!

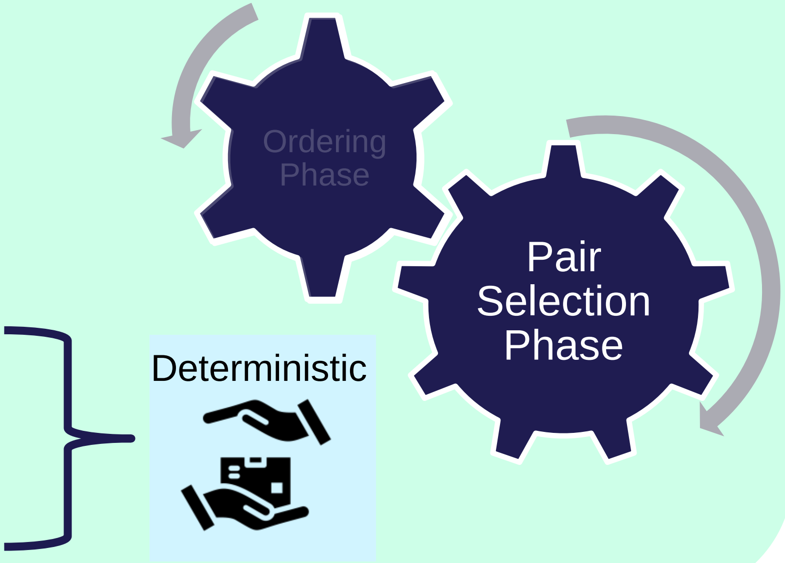
Each cycle, agents are assigned a 'DOCS ID' for partnership coordination and ordering.



LAMDLS-2

□ Coordinate changes within a coalition; keep neighbors idle.

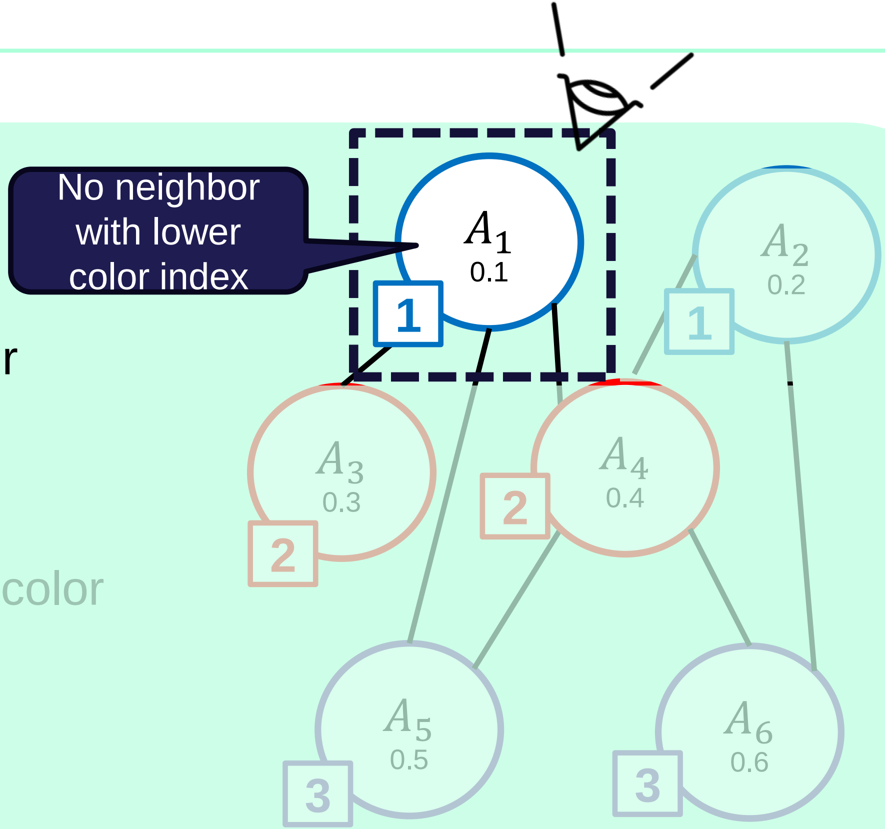
- Agents can offer or receive a coalition request.
- **Offer**: Share all local information with potential partner.
- **Receive**: Change value in a bilateral manner.



LAMDLS-2

Offer **Receiver**

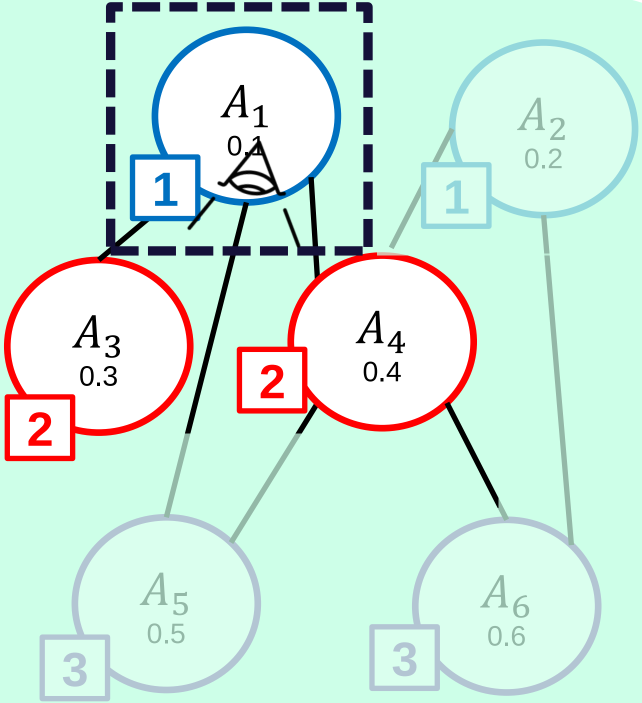
- When? All neighbors with a lower color index have selected their assignments.
- Who? Offer to a neighbor with a color index larger by 1.
- Tie breaker? Smaller DOCS id



LAMDLS-2

Offer **Receiver**

- When? All neighbors with a lower color index have selected their assignments.
- Who? Offer to a neighbor with a color index larger by 1.
- Tie breaker? Smaller DOCS id

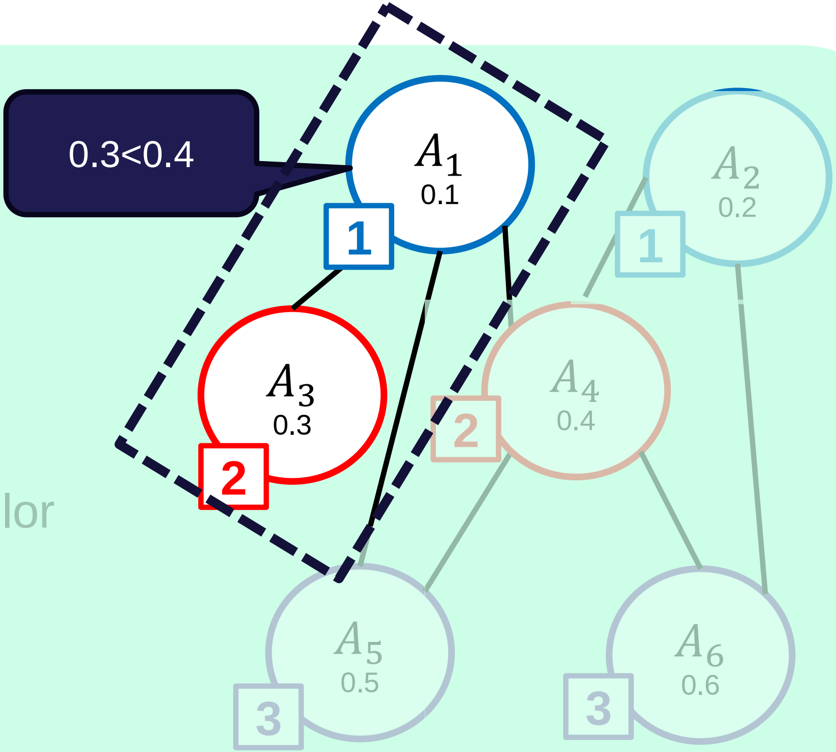


LAMDLS-2

Offer

Receiver

- When? All neighbors with a lower color index have selected their assignments.
- Who? Offer to a neighbor with a color index larger by 1.
- Tie breaker? Smaller DOCS id



LAMDLS-2

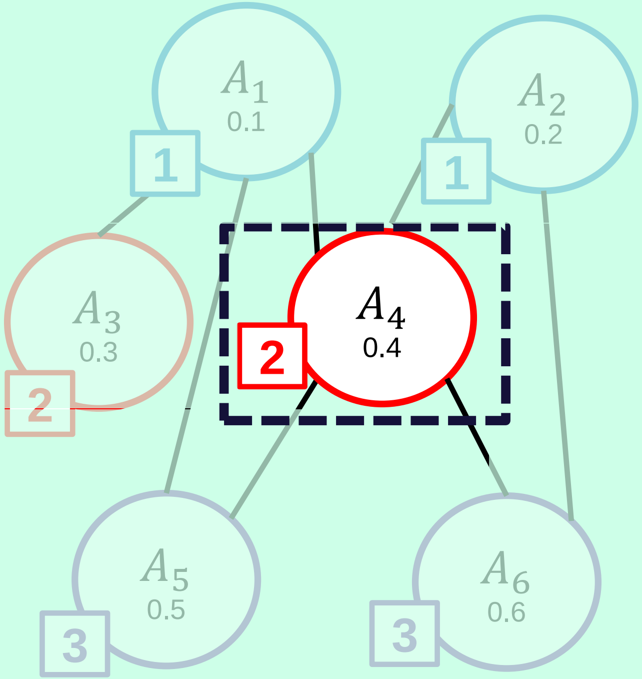
Offer

Receiver

- How? Commit to a neighbor, and if an offer is received, find bilateral values.

- Who? Wait for an offer from a neighbor with a color index smaller by 1.

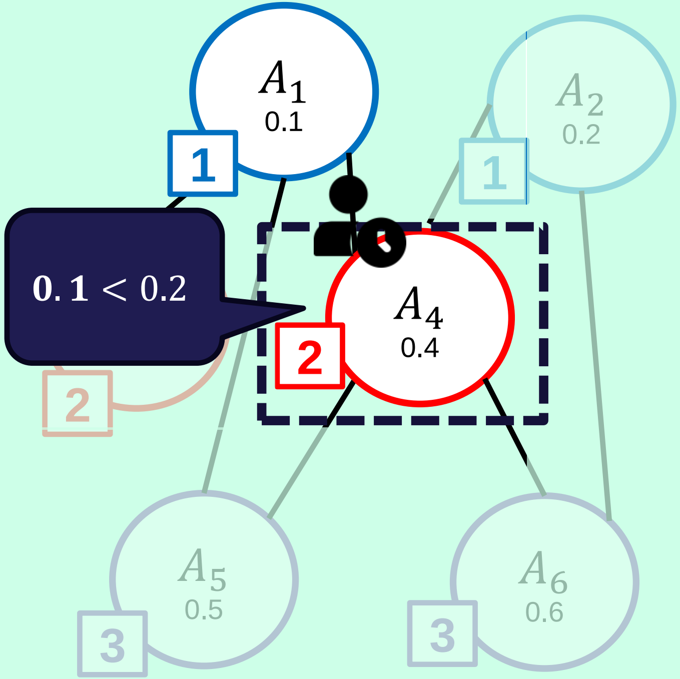
- Tie braker? DOCS id



LAMDLS-2

Offer Receiver

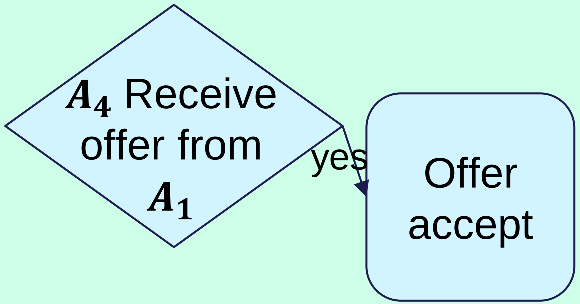
- How? Commit to a neighbor, and if an offer is received, find bilateral values.
- Who? Wait for an offer from a neighbor with a color index smaller by 1.
- Tie braker? DOCS id



LAMDLS-2

Offer

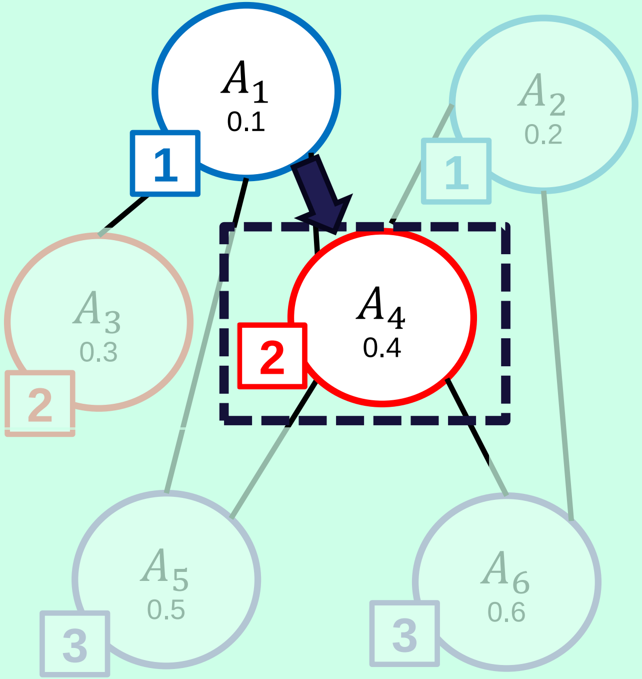
Receiver



A_4 Receive offer from A_1

yes

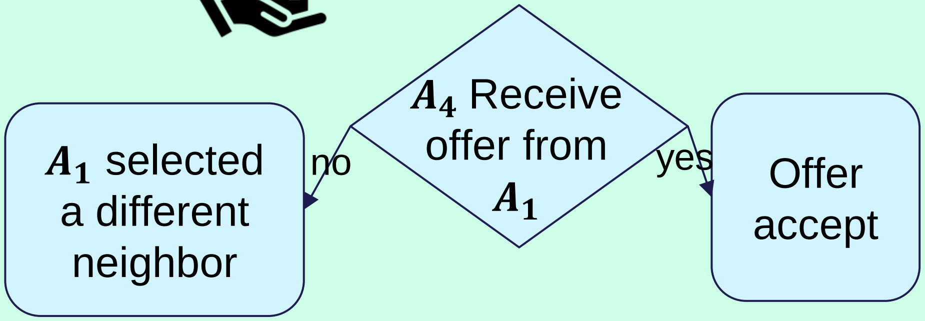
Offer accept



LAMDLS-2

Offer

Receiver



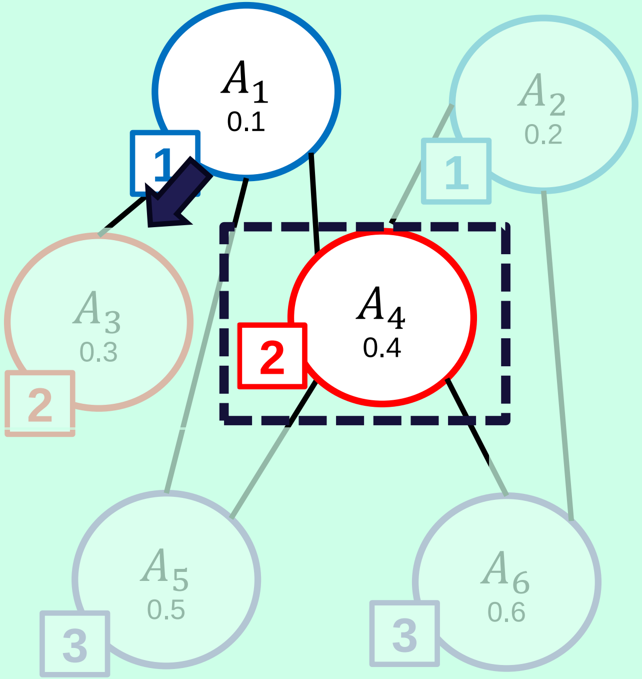
A_4 Receive offer from A_1

no

yes

A_1 selected a different neighbor

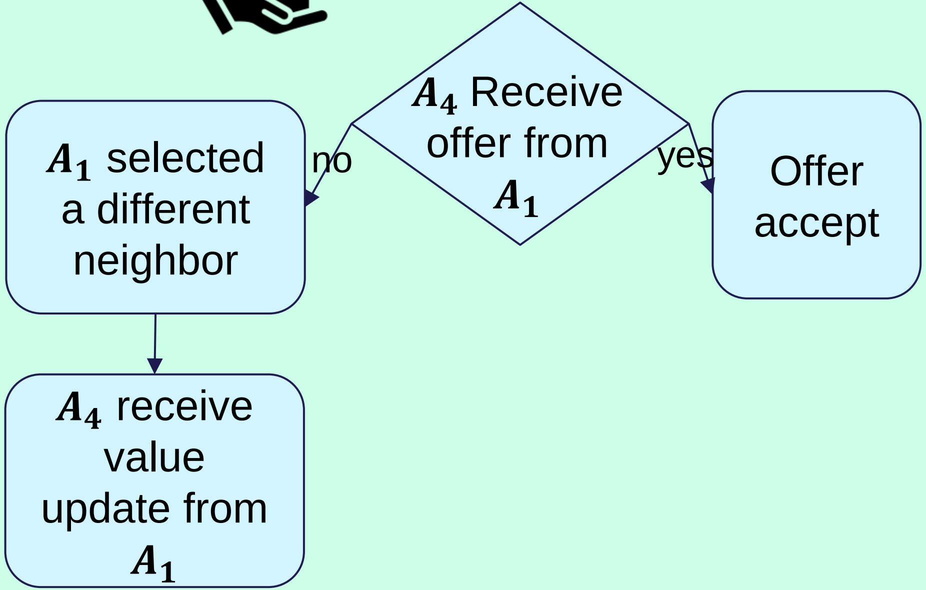
Offer accept

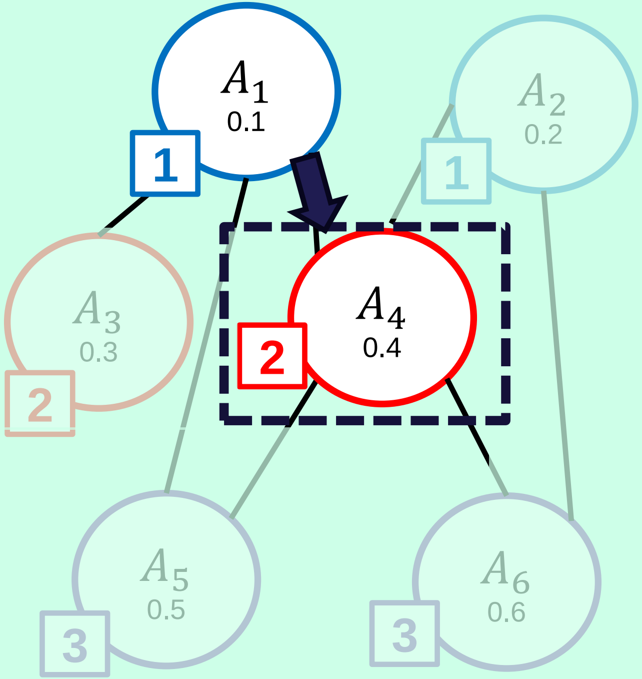


LAMDLS-2

Offer

Receiver

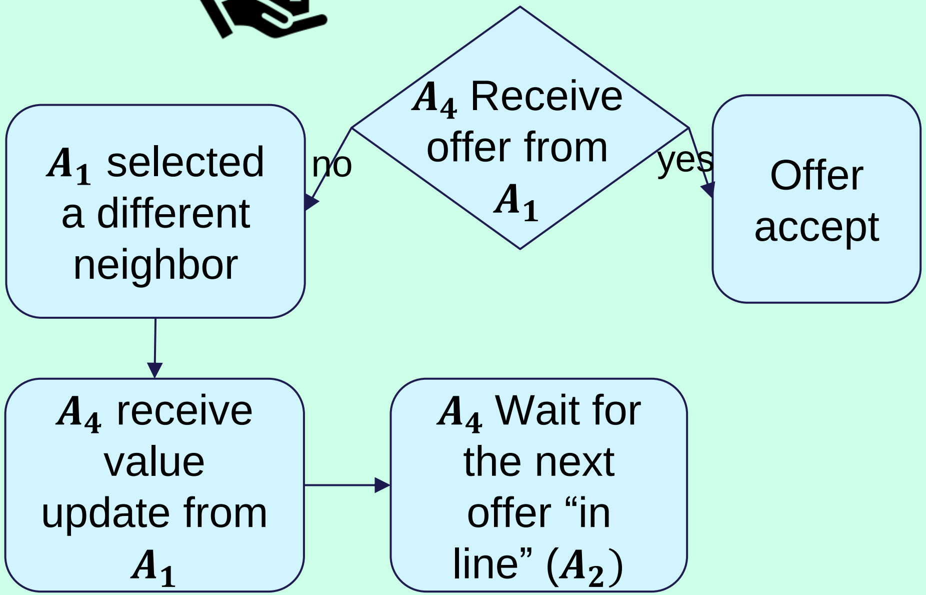

A_4 Receive offer from A_1
no
yes
A_1 selected a different neighbor
Offer accept
A_4 receive value update from A_1



LAMDLS-2

Offer

Receiver


A_4 Receive offer from A_1

no

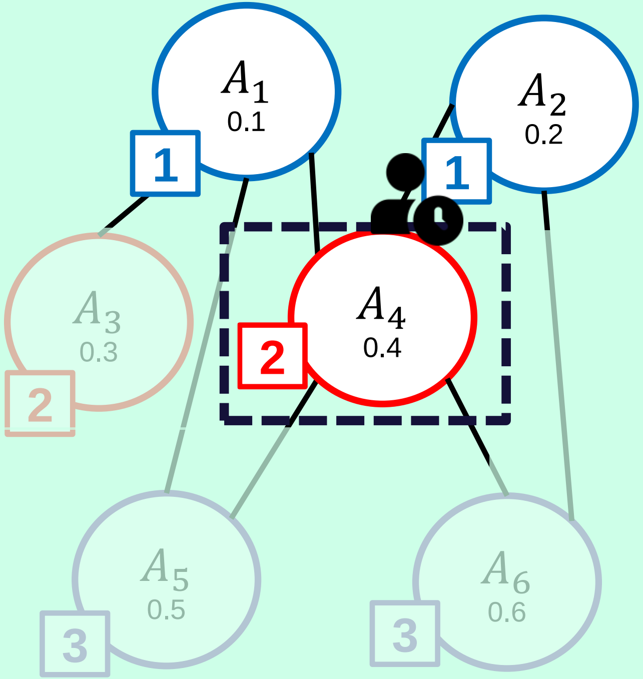
yes

A_1 selected a different neighbor

Offer accept

A_4 receive value update from A_1

A_4 Wait for the next offer “in line” (A_2)

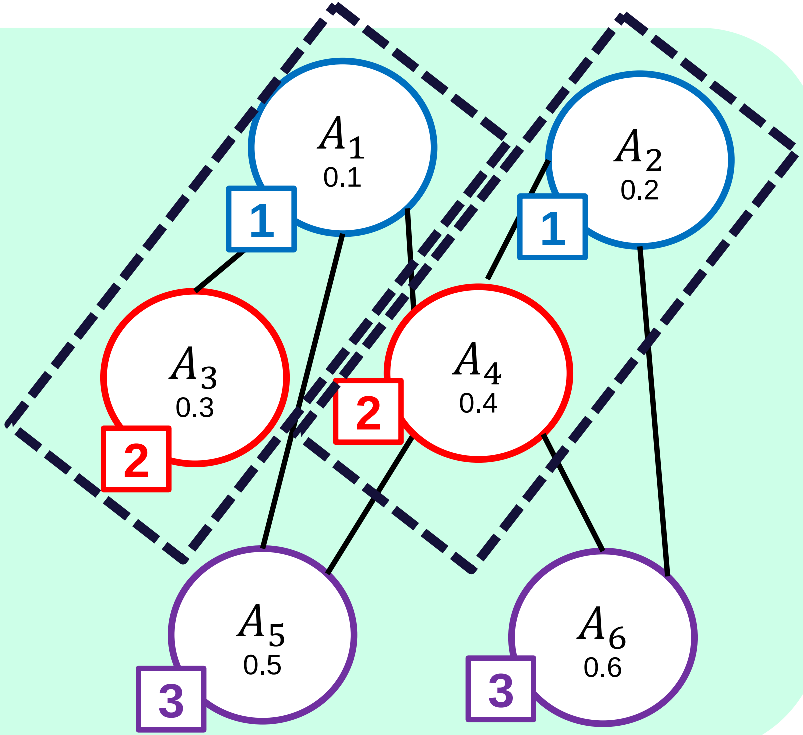


LAMDLS-2

Offer

Receiver

Pairs are selected deterministically without negotiation

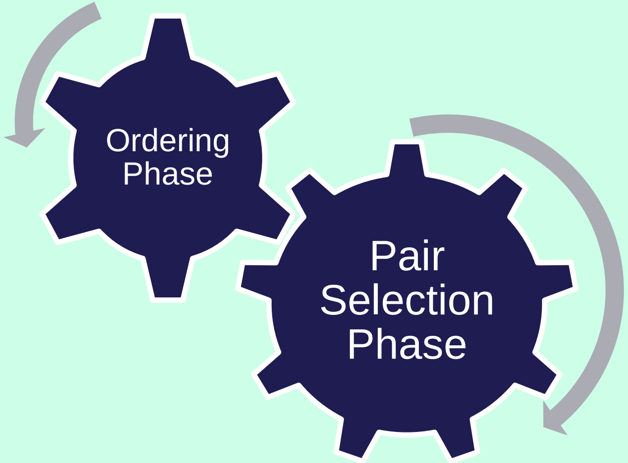


LAMDLS-2

Challenges!

□ Form all possible combinations of coalitions.

☑ Coordinate changes within a coalition; keep neighbors idle.

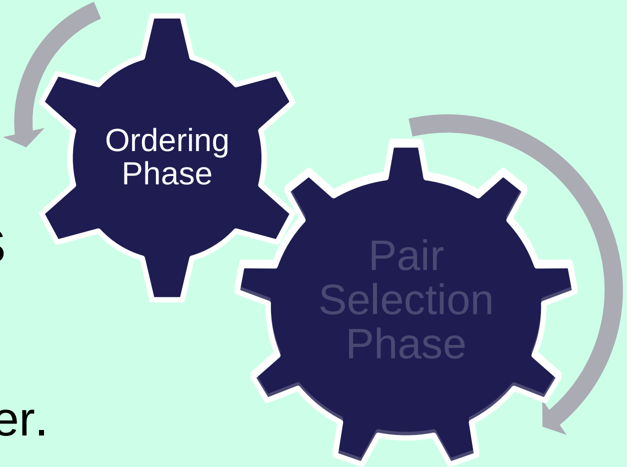


LAMDLS-2

Challenges!

□ Form all possible combinations of coalitions

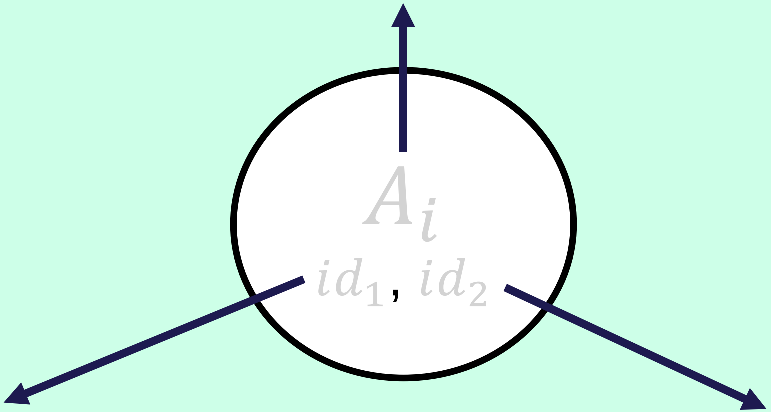
- Agents use DOCS to select colors.
- Colors are chosen based on DOCS IDs.
- Each DOCS run creates a new order.



LAMDLS-2

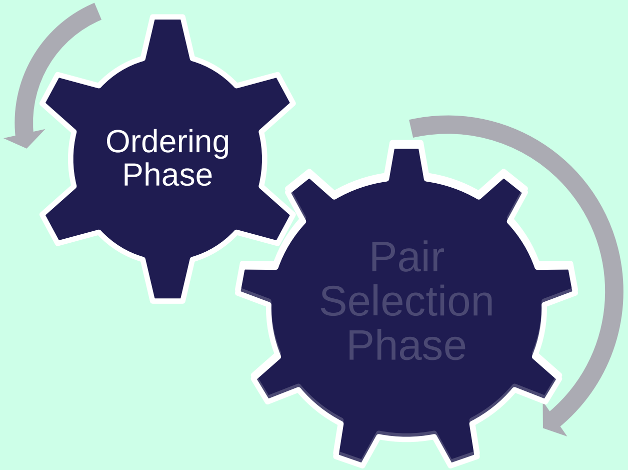
Example:

i – Agent’s id

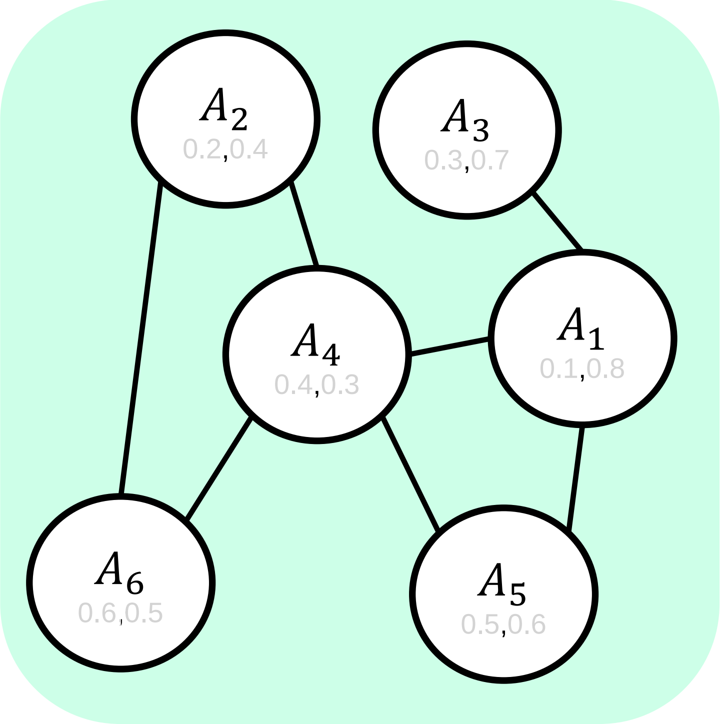


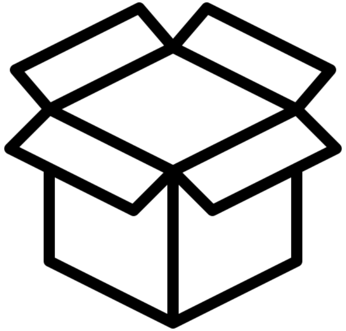
1st random DOCS id

2nd random DOCS id



LAMDLS-2

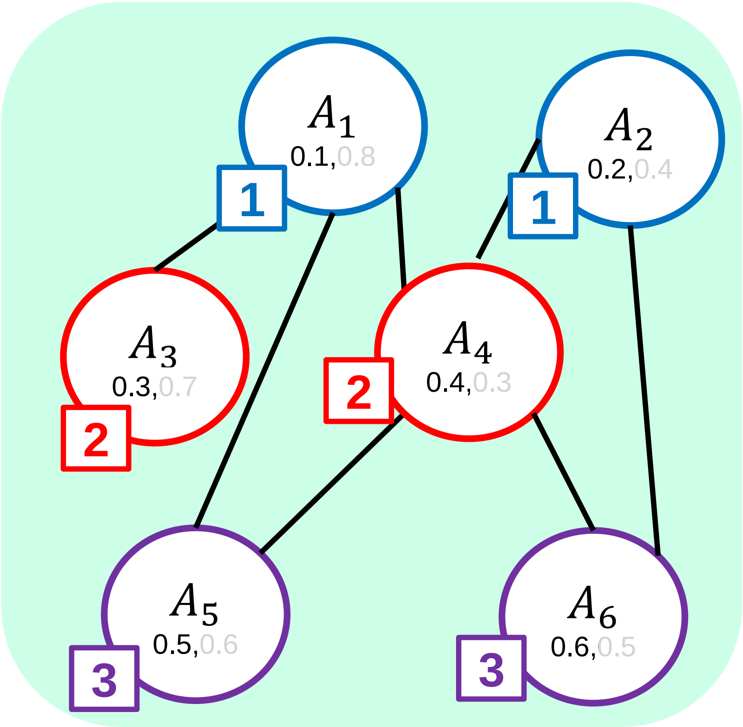




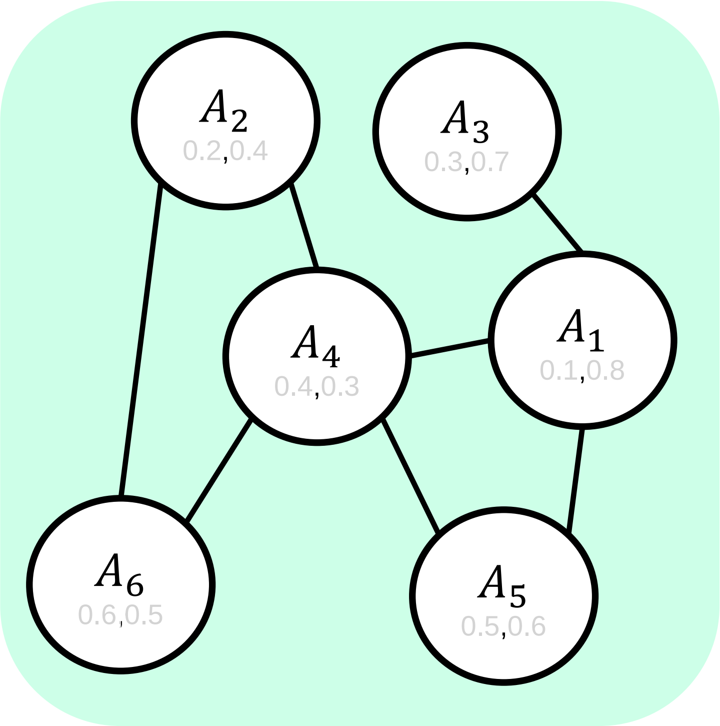
DOCS using
DOCS id 1

LAMDLS-2

DOCS using
DOCS id 1



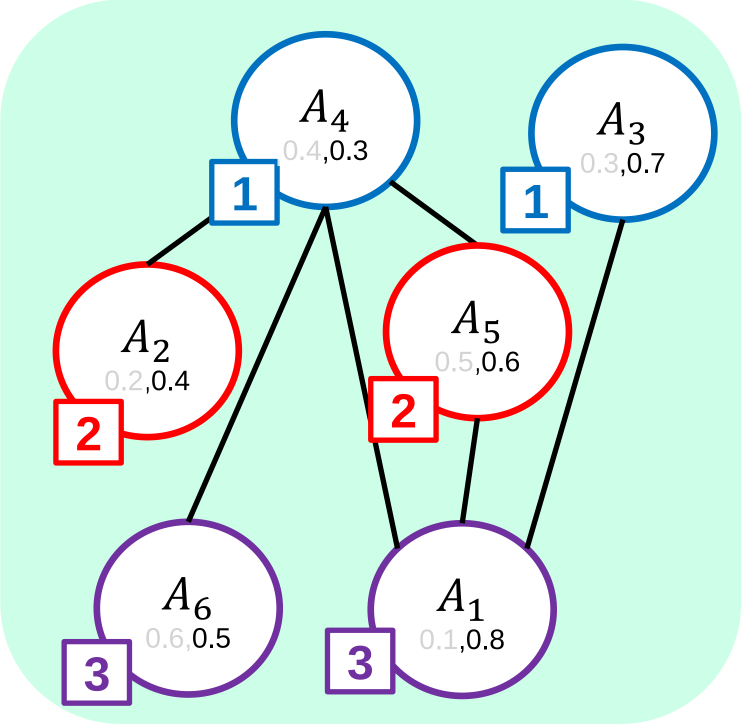
LAMDLS-2



DOCS using
DOCS id 2

LAMDLS-2

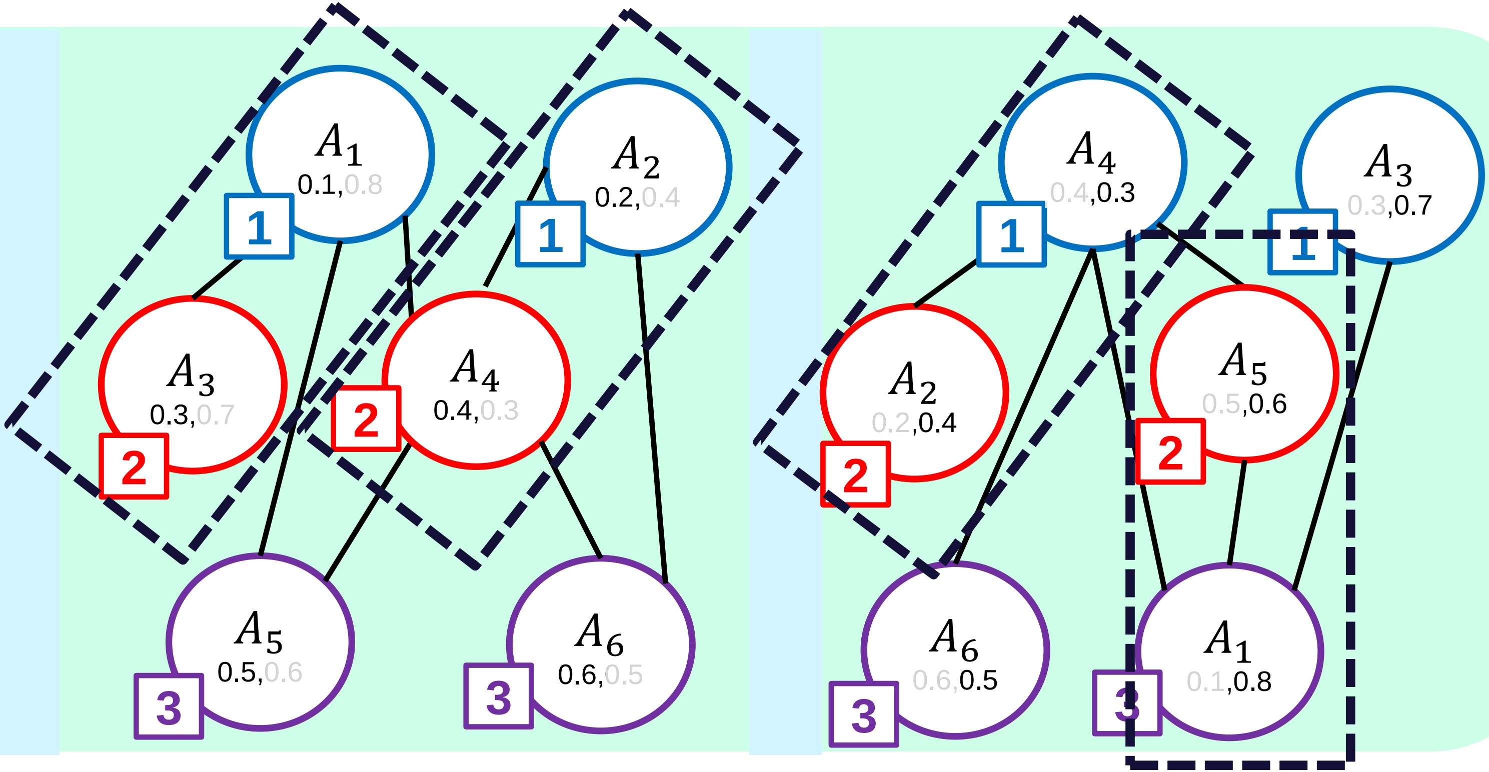
DOCS using DOCS id 2



LAMDLS-2

Different DOCS id

Different Order



Experimental Evaluation

Experimental Design

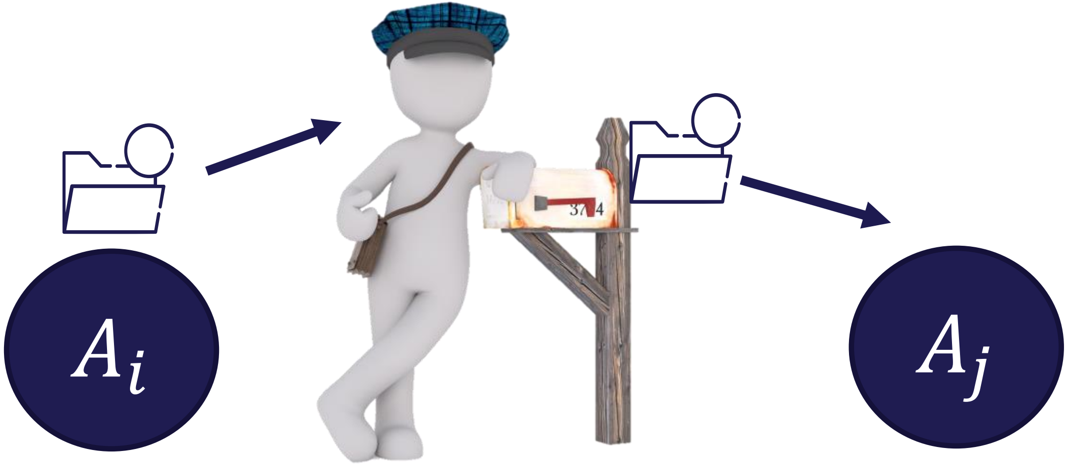
Simulator implemented by Java threads

All messages go through a “mailing agent” – simulates the delivery of messages

$$td_e \sim U(0,\boxed{UB})$$

$$td_e \sim Pois(\,|MSG|) * \boxed{m}$$

UB and **m** monitors the latency magnitude

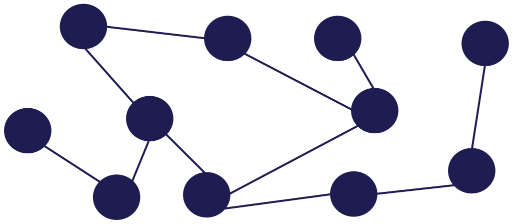


(Zivan and Meisels, 2006)

Experimental Design

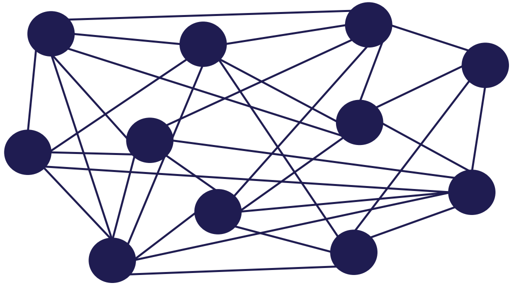
Sparse Random Uniform

(Gershman et al., 2009)



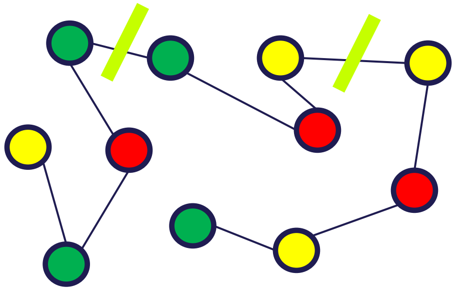
Dense Random Uniform

(Gershman et al., 2009)



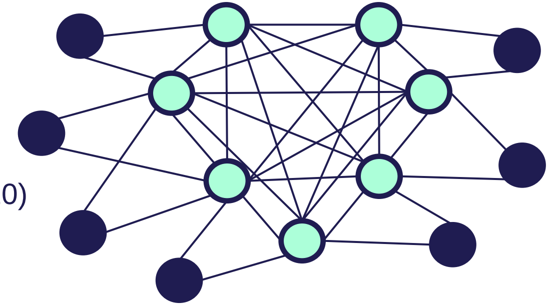
Graph Coloring

(Zivan et al., 2014)



Scale Free Network

(Kiekintveld et al., 2010)



Experimental Design

$$|A| = 50$$

50 Agents participation

Experimental Design

$|A| = 50$

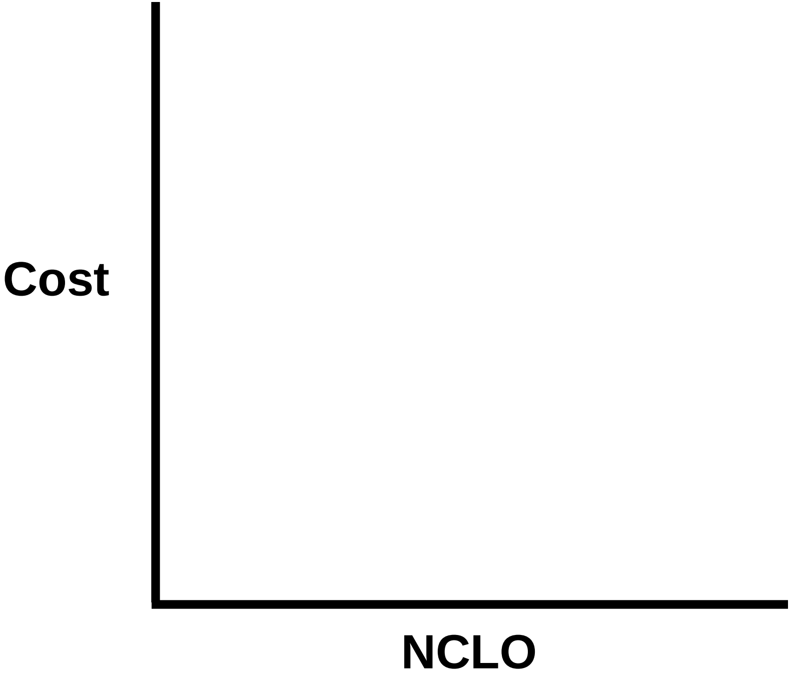
50 Agents participation

100 repetitions for each type problem

Experimental Evaluation

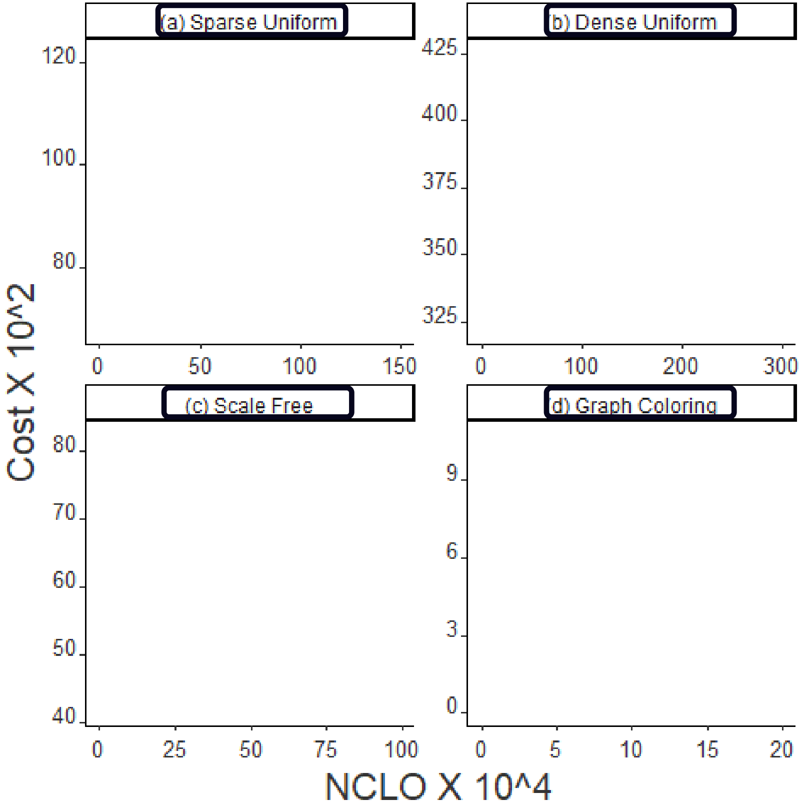
Y = Global aggregation of constraints' cost

Bird eye POV



X = **N**on-**C**oncurrent **L**ogic **O**perations: counts the algorithm's constraints check

Experimental Evaluation



$td_e \sim U(0, UB)$

$td_e \sim Pois(\,|MSG|) * m$

Algorithm (color)

LAMDLS-2

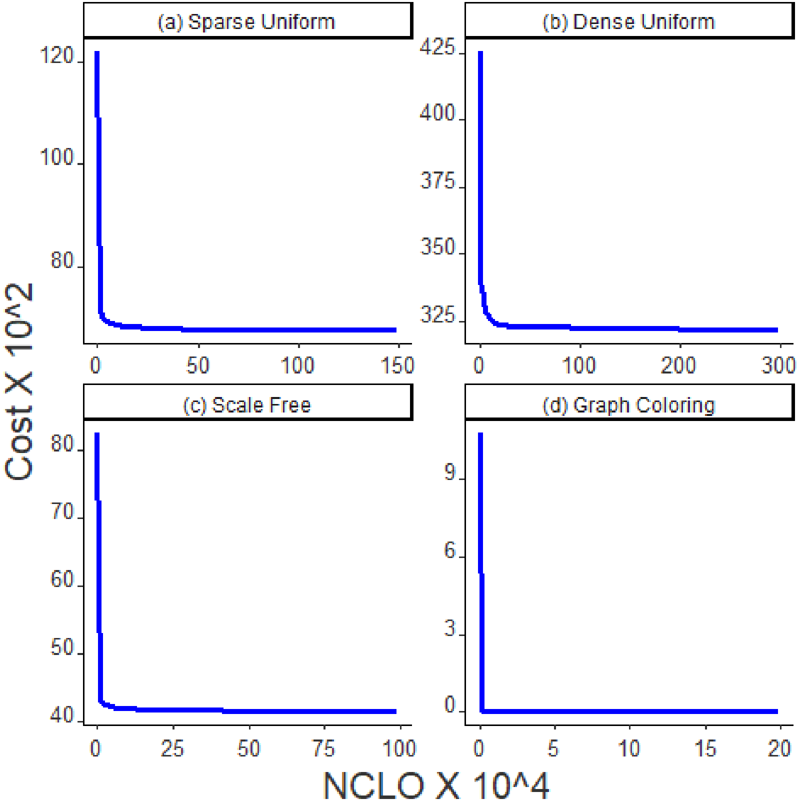
MGM-2

UB (line type)

$U(0{,}0)$ - no latency

$U(0{,}10K)$

Experimental Evaluation



$td_e \sim U(0, UB)$

$td_e \sim Pois(\,|MSG|) * m$

Algorithm (color)

🟦 LAMDLS-2

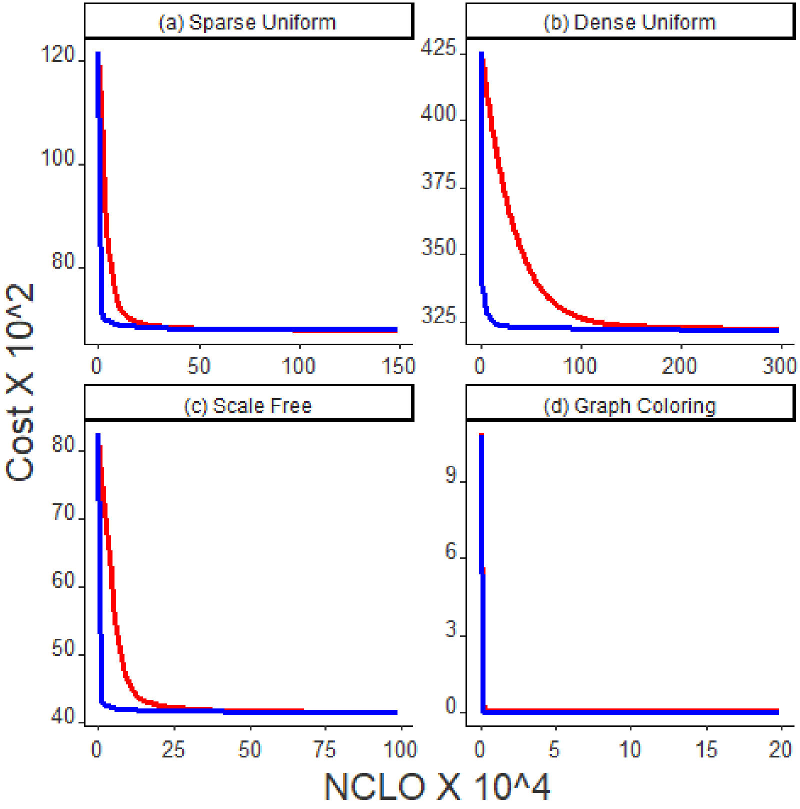
🟥 MGM-2

UB (line type)

▬ $U(0{,}0)$ - no latency

┅ $U(0{,}10K)$

Experimental Evaluation



$td_e \sim U(0, UB)$

$td_e \sim Pois(|MSG|) * m$

Algorithm (color)

🟦 LAMDLS-2

🟥 MGM-2

UB (line type)

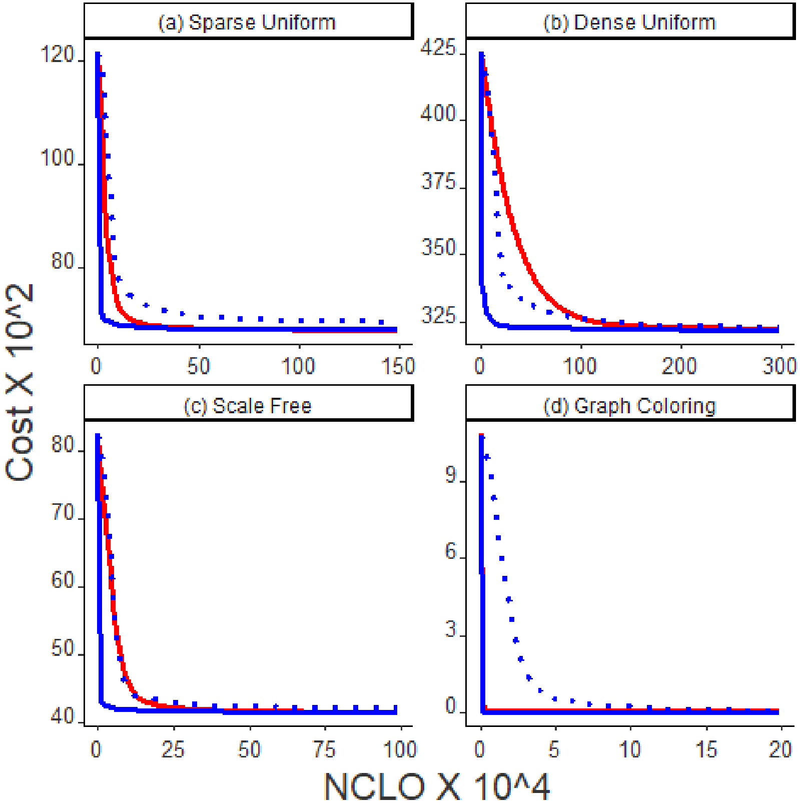
▬ $U(0,0)$ - no latency

╍ $U(0,10K)$

Experimental Evaluation

$td_e \sim U(0, UB)$

$td_e \sim Pois(\,|MSG|) * m$



Algorithm (color)

🟦 LAMDLS-2

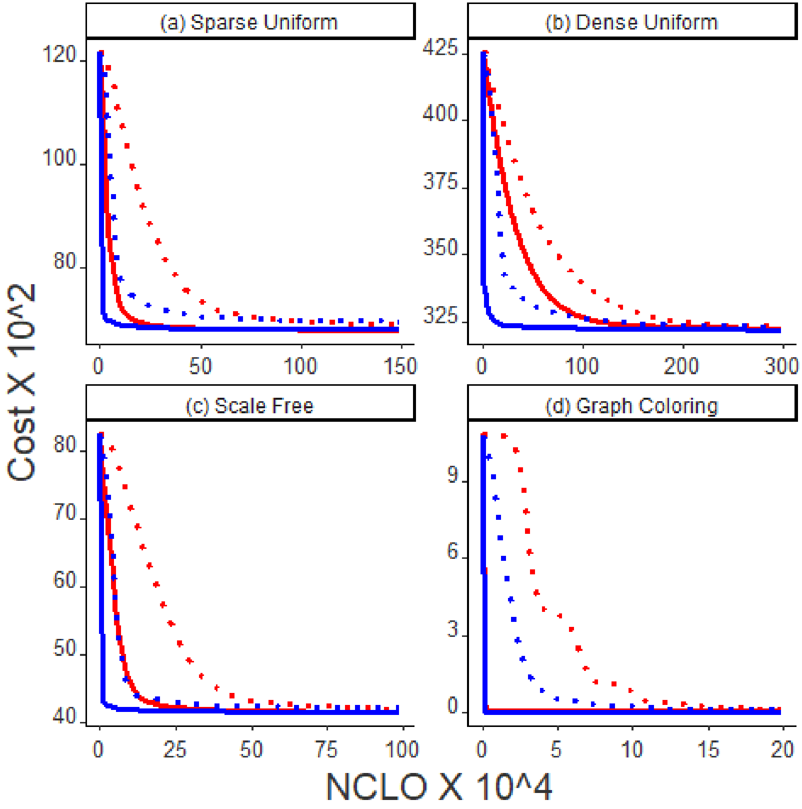
🟥 MGM-2

UB (line type)

▬ $U(0,0)$ - no latency

▮▮▮ $U(0,10K)$

Experimental Evaluation



$td_e \sim U(0, UB)$

$td_e \sim Pois(\,|MSG|) * m$

Algorithm (color)

🟦 LAMDLS-2

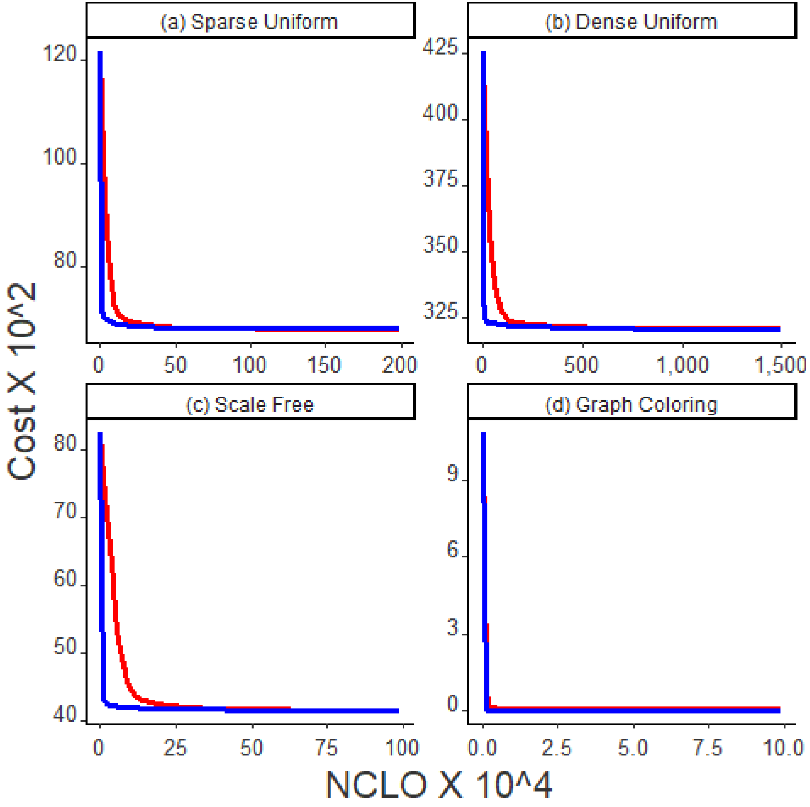
🟥 MGM-2

UB (line type)

▬ $U(0,0)$ - no latency

┅ $U(0,10K)$

Experimental Evaluation



$td_e \sim U(0,UB)$

$td_e \sim Pois(\,|MSG|) * m$

Algorithm (color)

🟦 LAMDLS-2

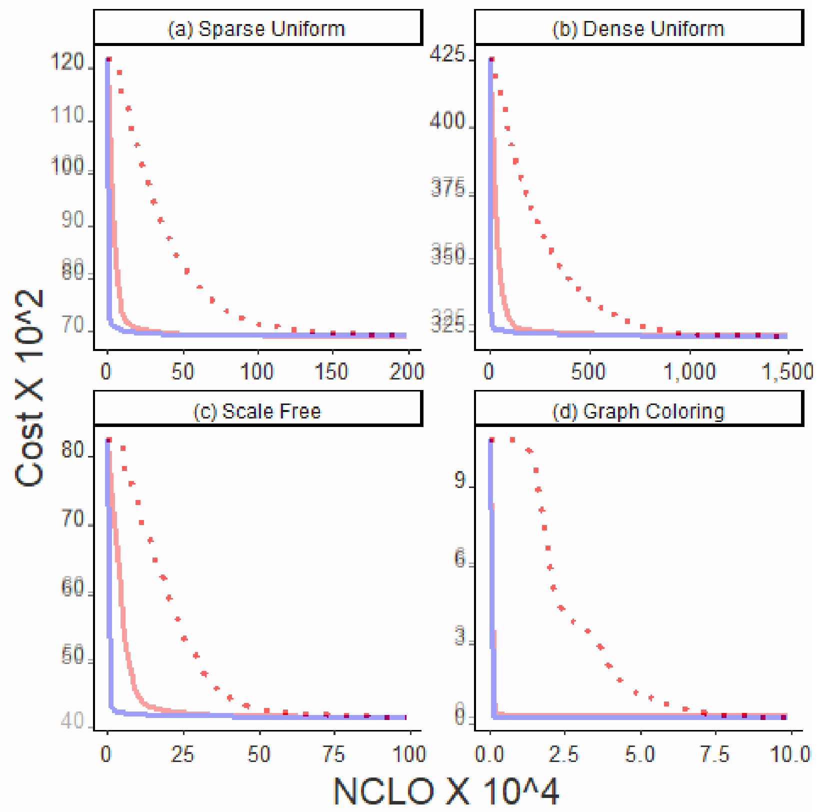
🟥 MGM-2

m (line type)

▬ $Pois((\,|MSG|) * 0$ - no latency

┅ $Pois((\,|MSG|) * 50$

Experimental Evaluation



$td_e \sim U(0,UB)$

$td_e \sim Pois(|MSG|) * m$

Algorithm (color)

🟦 LAMDLS-2

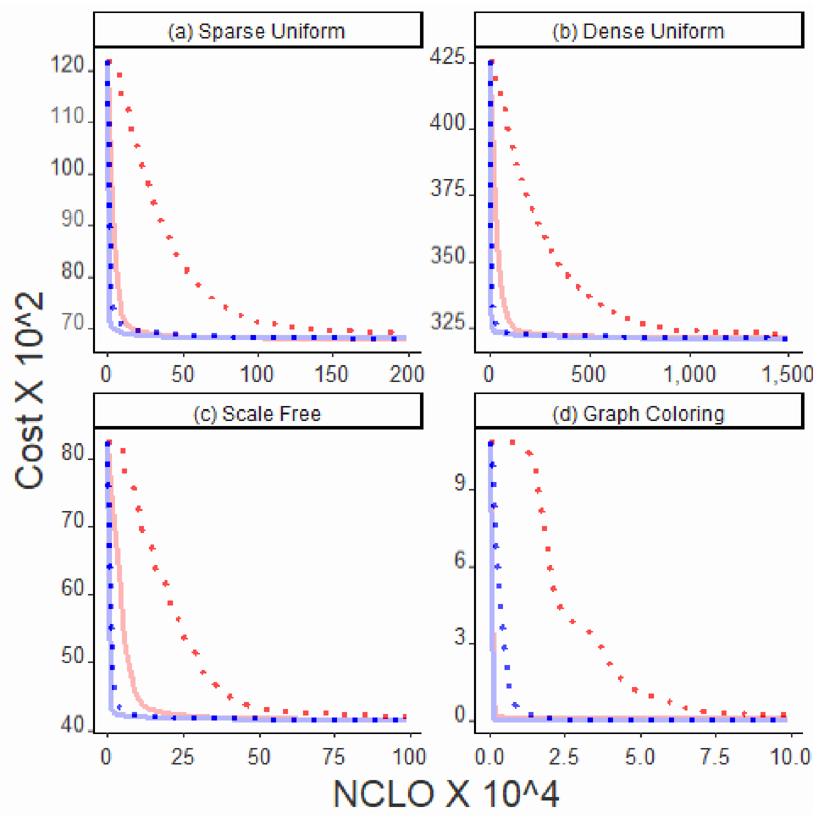
🟥 MGM-2

m (line type)

▬ $Pois((|MSG|) * 0$ - no latency

▮▮▮ $Pois((|MSG|) * 50$

Experimental Evaluation



$td_e \sim U(0,UB)$

$td_e \sim Pois(\,|MSG|) * m$

Algorithm (color)

🟦 LAMDLS-2

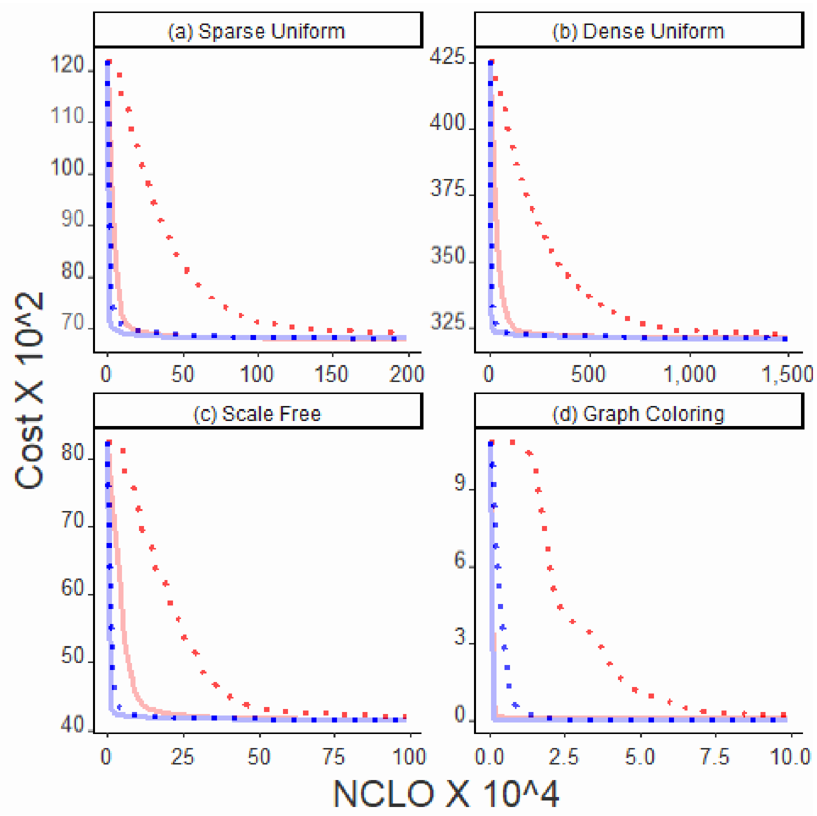
🟥 MGM-2

m (line type)

▬ $Pois((\,|MSG|) * 0$ - no latency

▮▮▮ $Pois((\,|MSG|) * 50$

Experimental Evaluation



$td_e \sim U(0,UB)$

$td_e \sim Pois(\,|MSG|) * m$

Algorithm (color)

🟦 LAMDLS-2

🟥 MGM-2

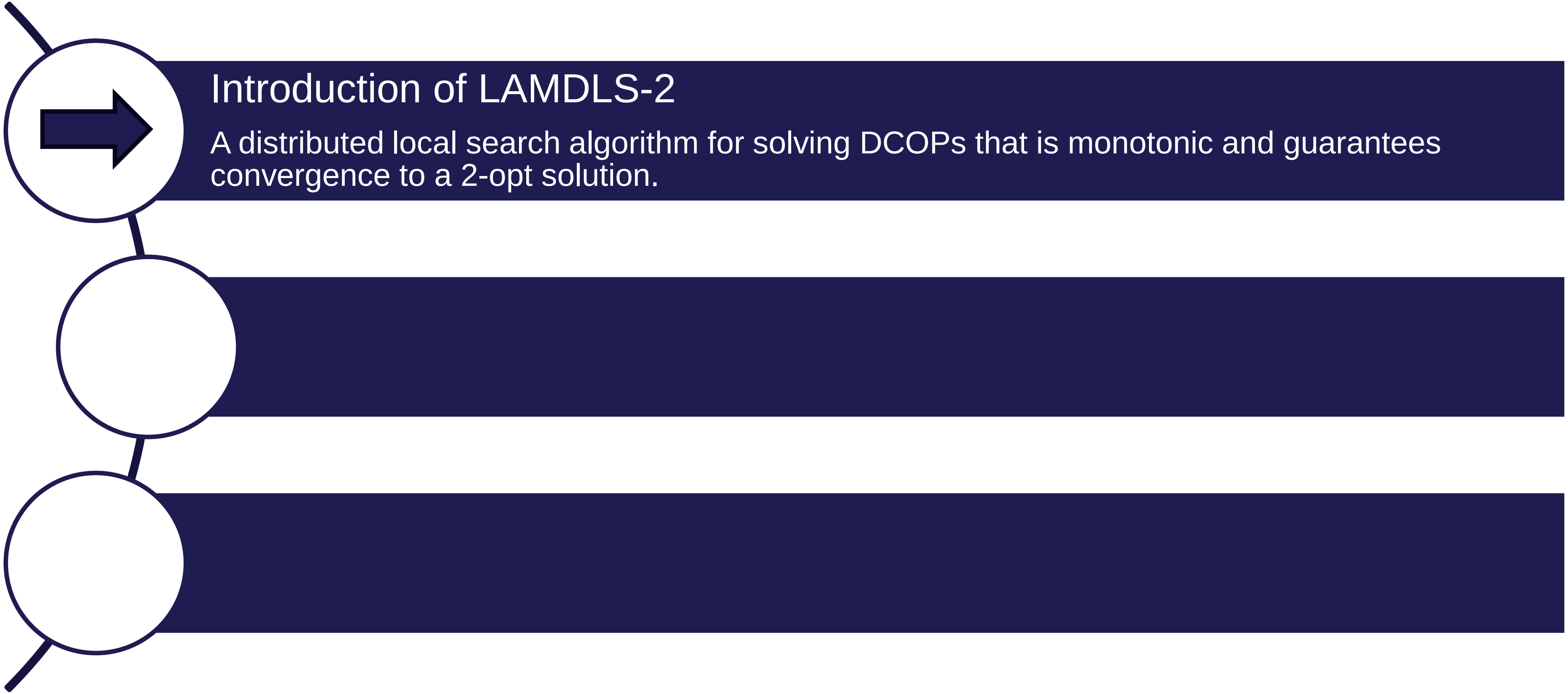
m (line type)

▬ $Pois((\,|MSG|) * 0$ - no latency

┅ $Pois((\,|MSG|) * 50$

Conclusions

Conclusions



Introduction of LAMDLS-2

A distributed local search algorithm for solving DCOPs that is monotonic and guarantees convergence to a 2-opt solution.

Conclusions

Introduction of LAMDLS-2

A distributed local search algorithm for solving DCOPs that is monotonic and guarantees convergence to a 2-opt solution.

Comparison to MGM-2

LAMDLS-2 converges faster and uses the communication network more efficiently than MGM-2, with agents spending less idle time.

Conclusions

Introduction of LAMDLS-2

A distributed local search algorithm for solving DCOPs that is monotonic and guarantees convergence to a 2-opt solution.

Comparison to MGM-2

LAMDLS-2 converges faster and uses the communication network more efficiently than MGM-2, with agents spending less idle time.

Suitability for Realistic Scenarios

LAMDLS-2 is particularly effective in scenarios with message delays.

Paper’s QR code

THANK YOU!

Ben Rachmut
rachmut@post.bgu.ac.il

Any Questions