AAAI-20 Tutorial on:

Multi-Agent Distributed Constrained Optimization

Ferdinando Fioretto
Syracuse University

William Yeoh
Washington University in St. Louis

Schedule

- 2:00pm: Preliminaries
- 2:20pm: DCOP Algorithms
- 3:00pm: DCOP Extensions
- 3:20pm: Applications
- 3:35pm: Challenges and Open Questions
- 3:45pm: Coffee!!

Lil' Bit of Shameless Promotion :)

- Tutorial materials are based on our recent JAIR survey paper:

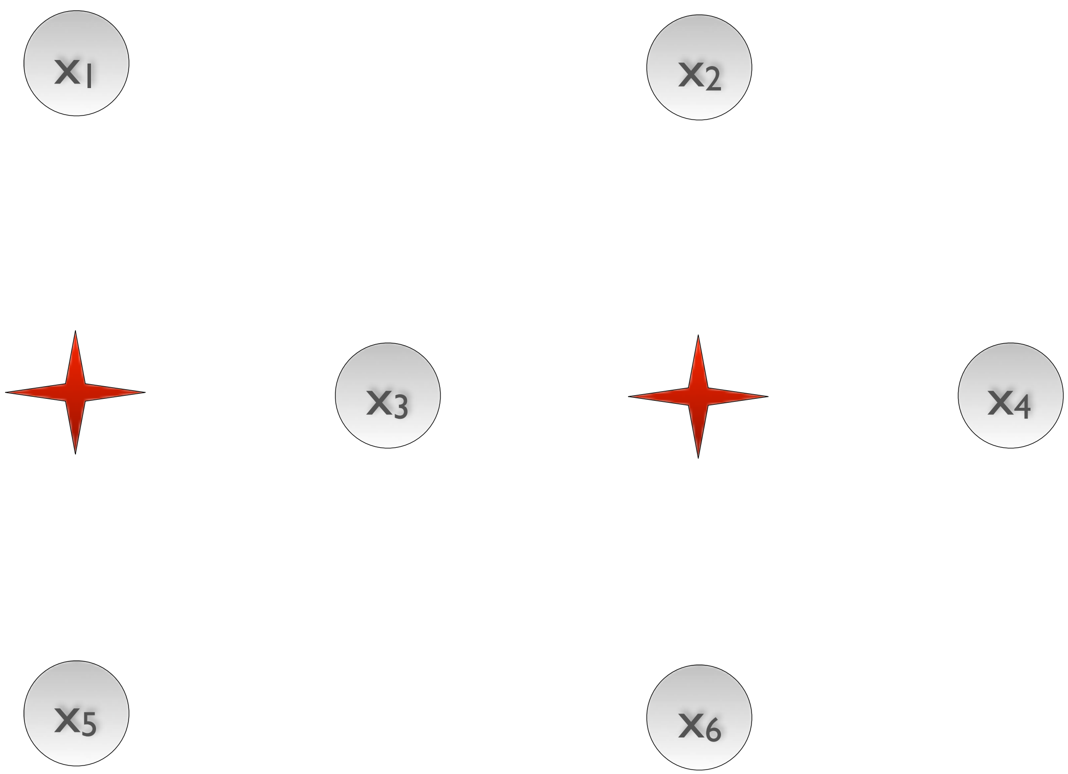
  Ferdinando Fioretto, Enrico Pontelli, and William Yeoh.
  Distributed Constraint Optimization Problems and Applications: A Survey.
  Journal of Artificial Intelligence Research (JAIR). 2018.

- Includes more models, algorithms, and applications.

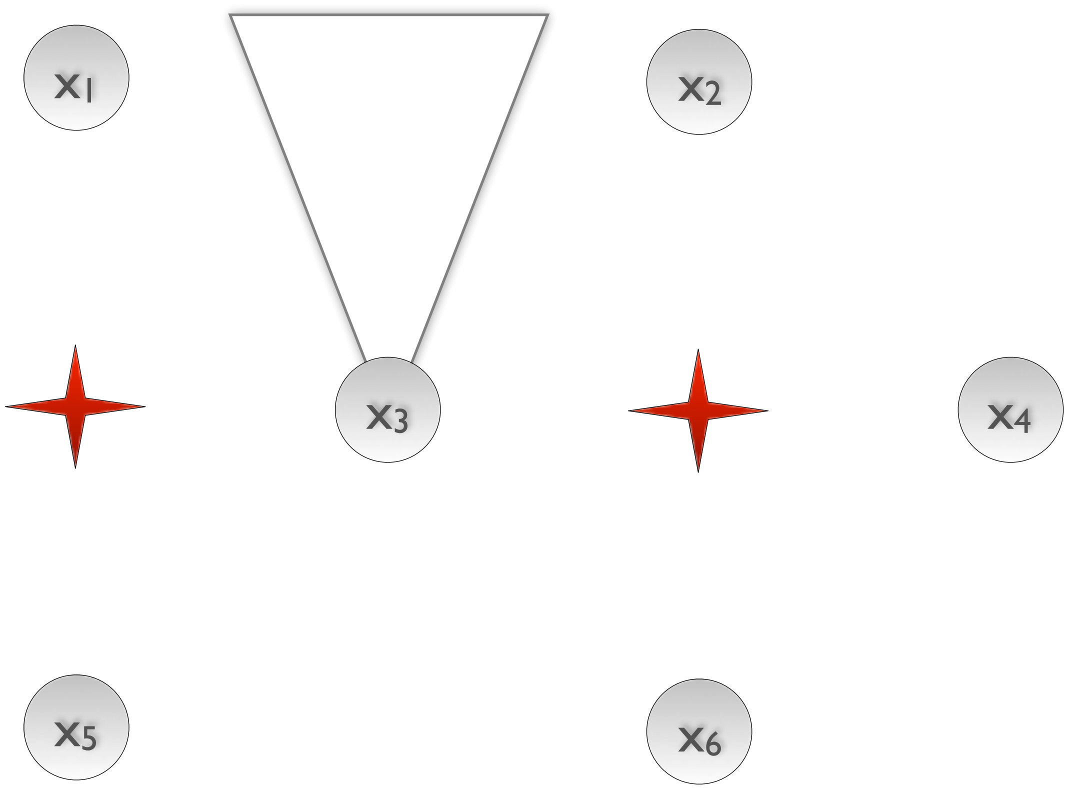
Preliminaries

AAAI-20 Tutorial on
Multi-Agent Distributed Constrained Optimization

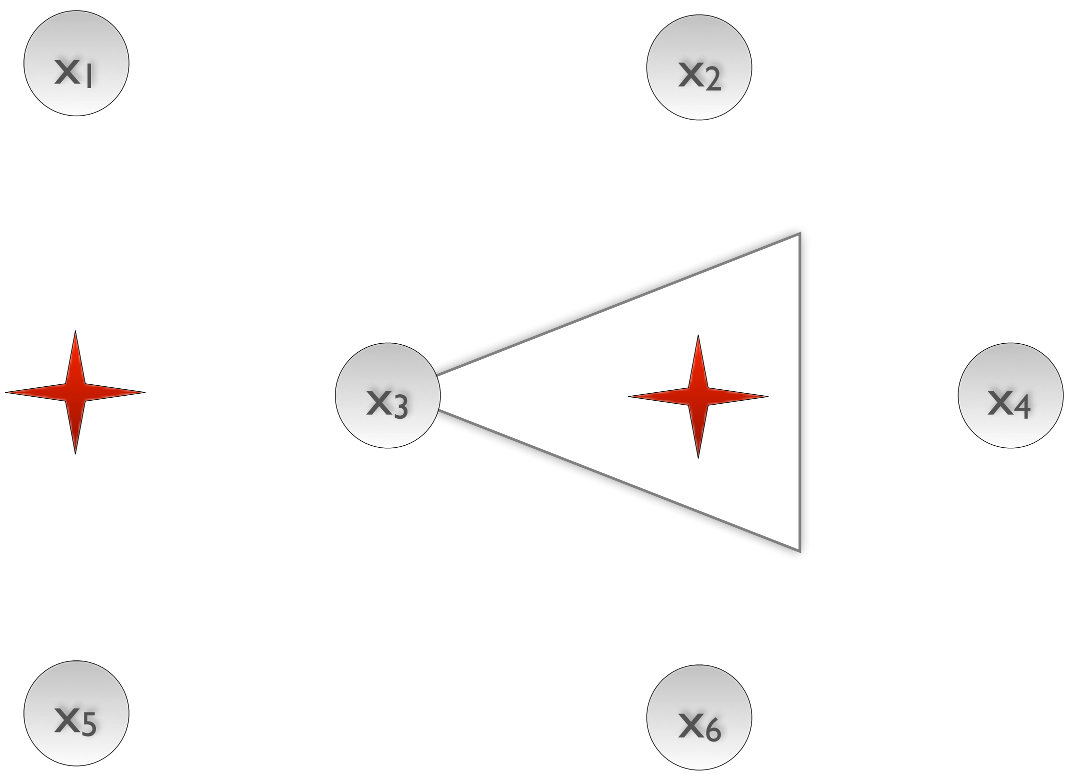
MOTIVATING DOMAIN: SENSOR NETWORK



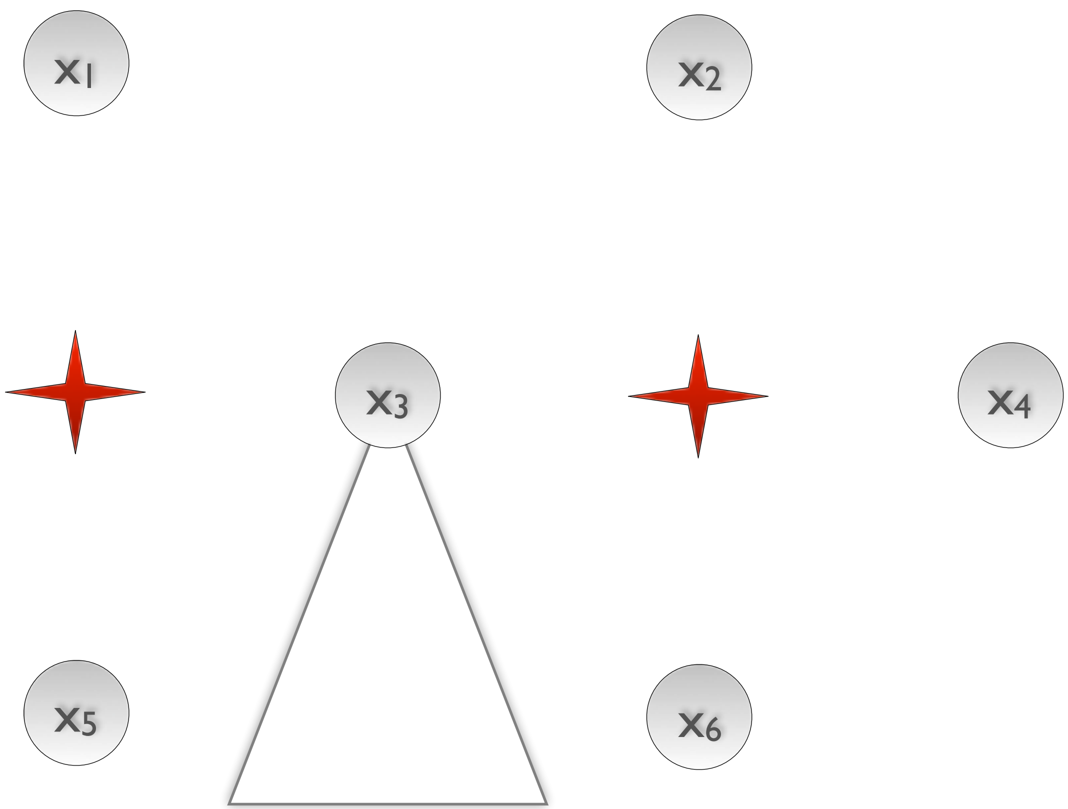
MOTIVATING DOMAIN: SENSOR NETWORK



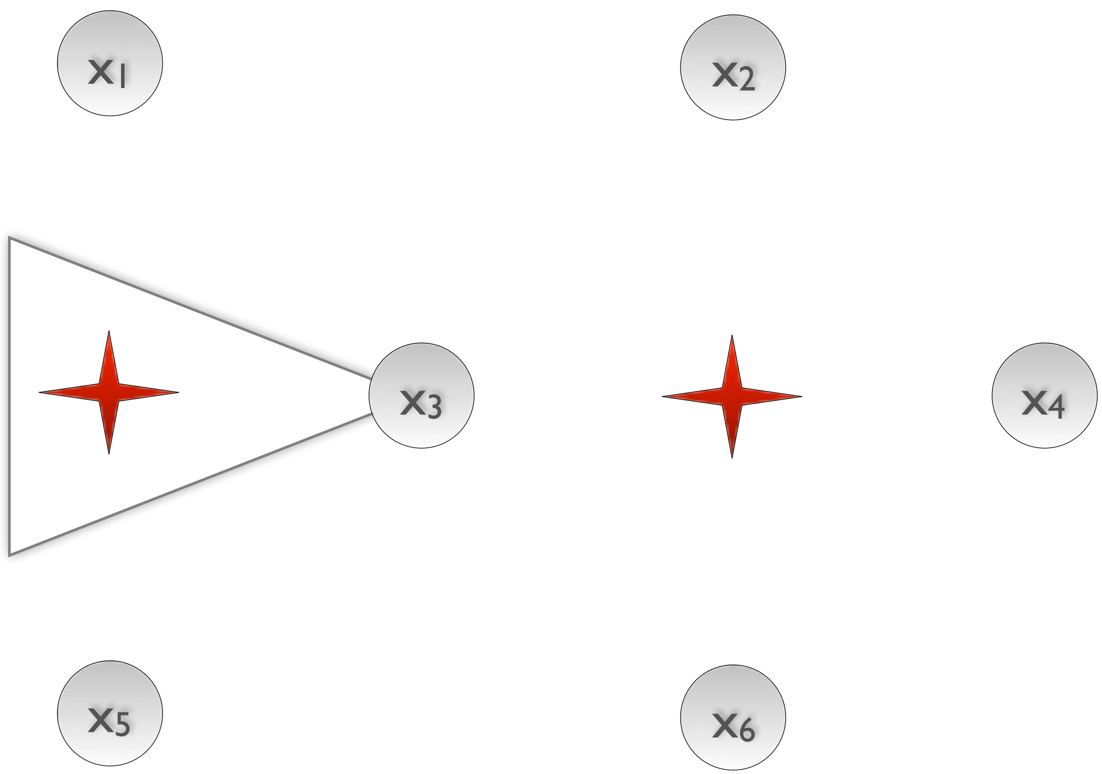
MOTIVATING DOMAIN: SENSOR NETWORK



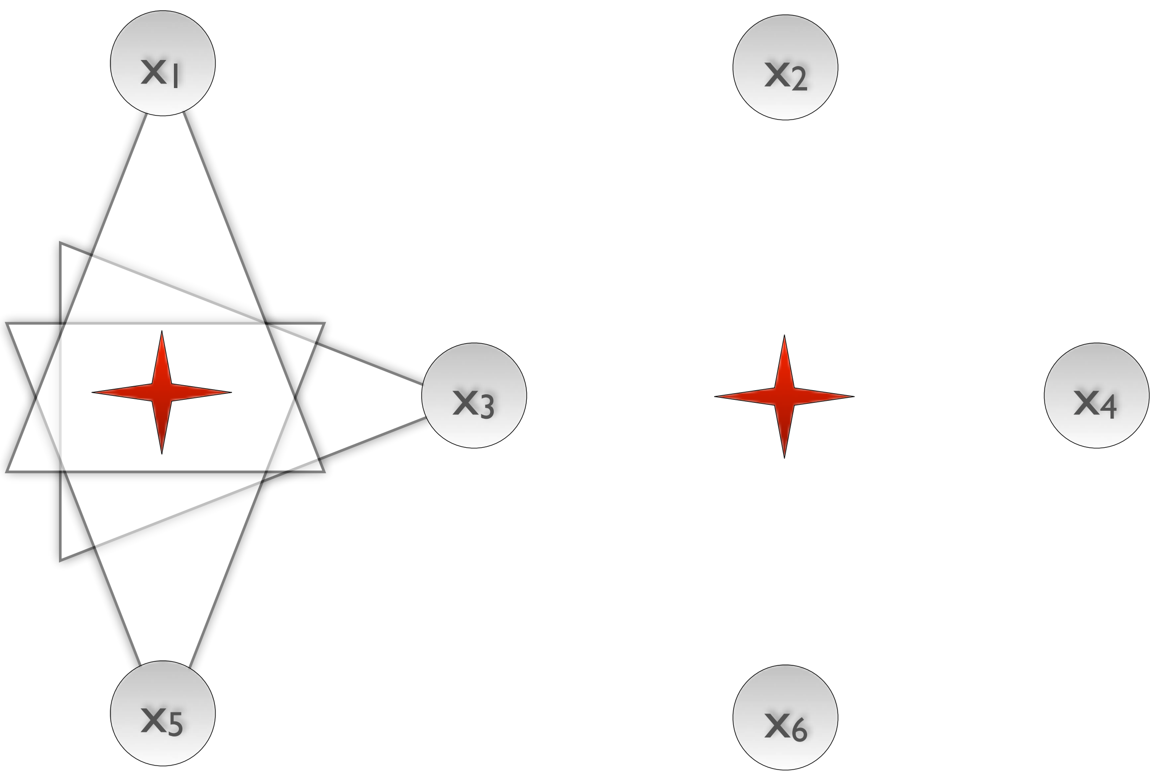
MOTIVATING DOMAIN: SENSOR NETWORK



MOTIVATING DOMAIN: SENSOR NETWORK



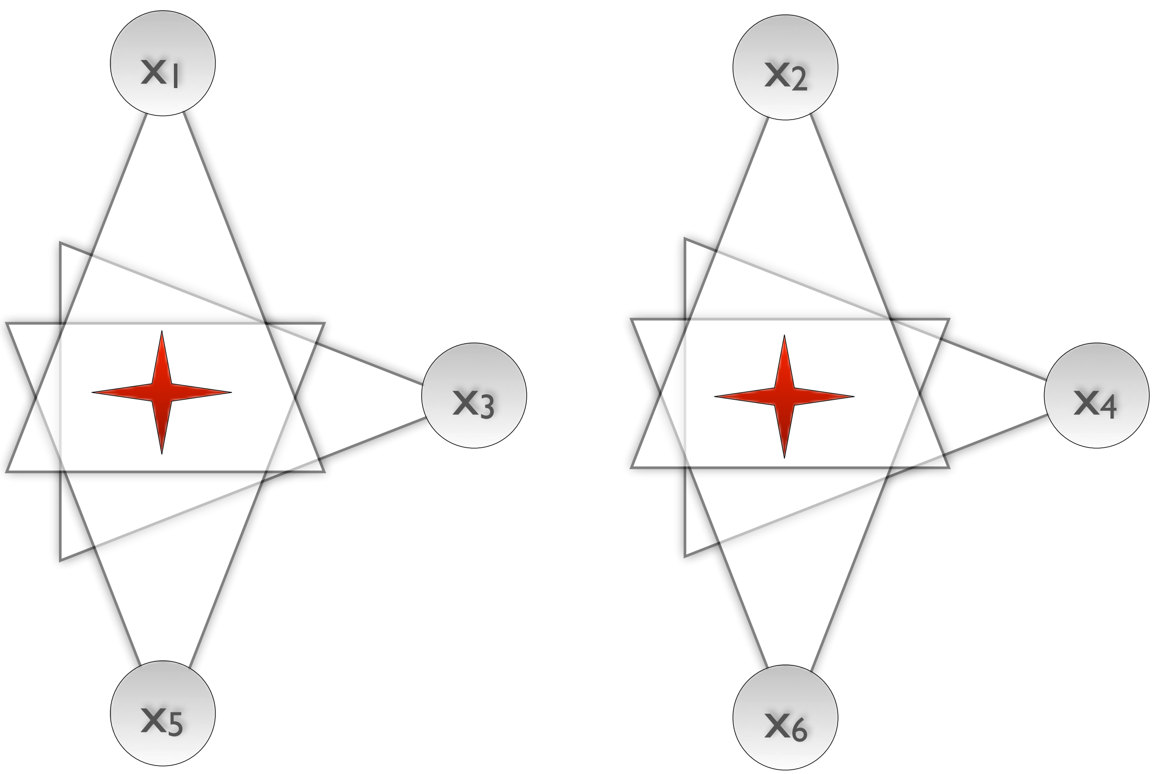
MOTIVATING DOMAIN: SENSOR NETWORK



x_1	x_3	x_5	Sat?
N	N	N	X
N	N	E	X
…			X
S	**W**	**N**	✓
…			X
W	W	W	X

Model the problem as a CSP

MOTIVATING DOMAIN: SENSOR NETWORK



x_1	x_3	x_5	Sat?
N	*N*	*N*	X
N	*N*	*E*	X
…			X
S	**W**	**N**	✓
…			X
W	*W*	*W*	X

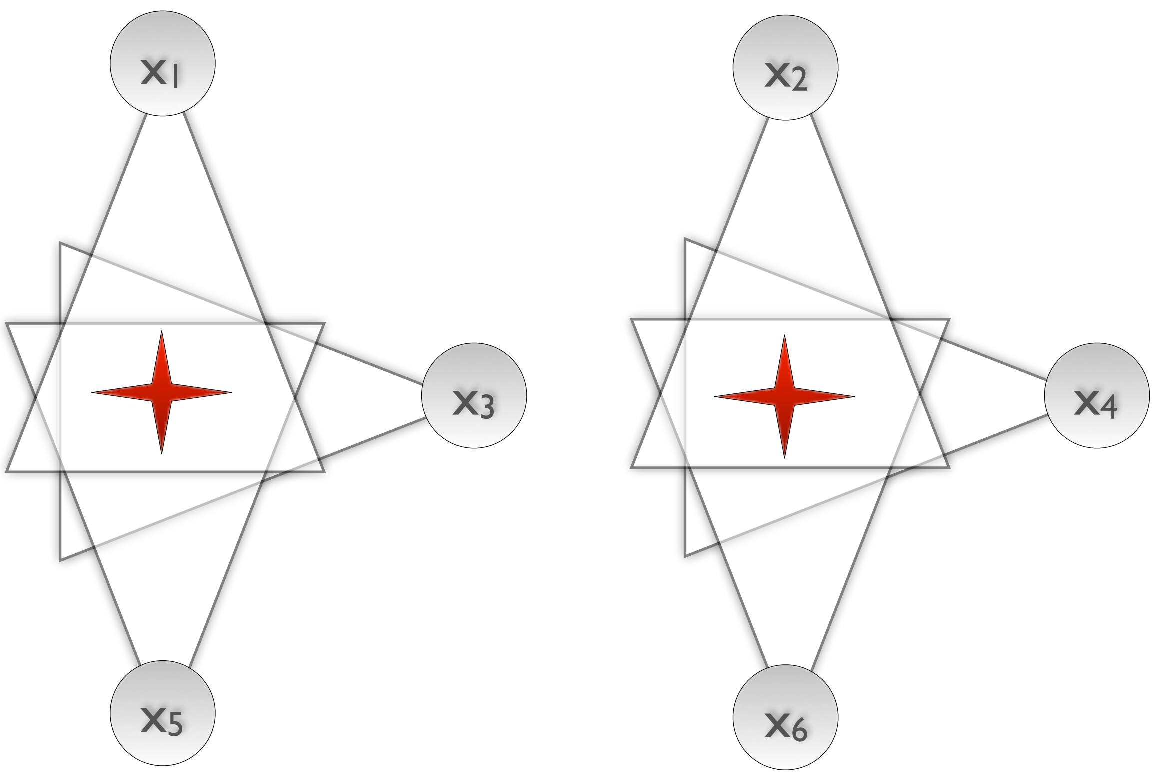
Model the problem as a CSP

CSP
CONSTRAINT SATISFACTION

- Variables $X = \{x_1, \ldots, x_n\}$
- Domains $D = \{D_1, \ldots, D_n\}$
- Constraints $C = \{c_1, \ldots, c_m\}$
  where a constraint $c_i \subseteq D_{i_1} \times D_{i_2} \times \ldots \times D_{i_n}$
  denotes the possible valid joint assignments for the variables $x_{i_1}, x_{i_2}, \ldots, x_{i_n}$ it involves

- **GOAL**: Find an assignment to all variables that satisfies all the constraints

CSP
CONSTRAINT SATISFACTION

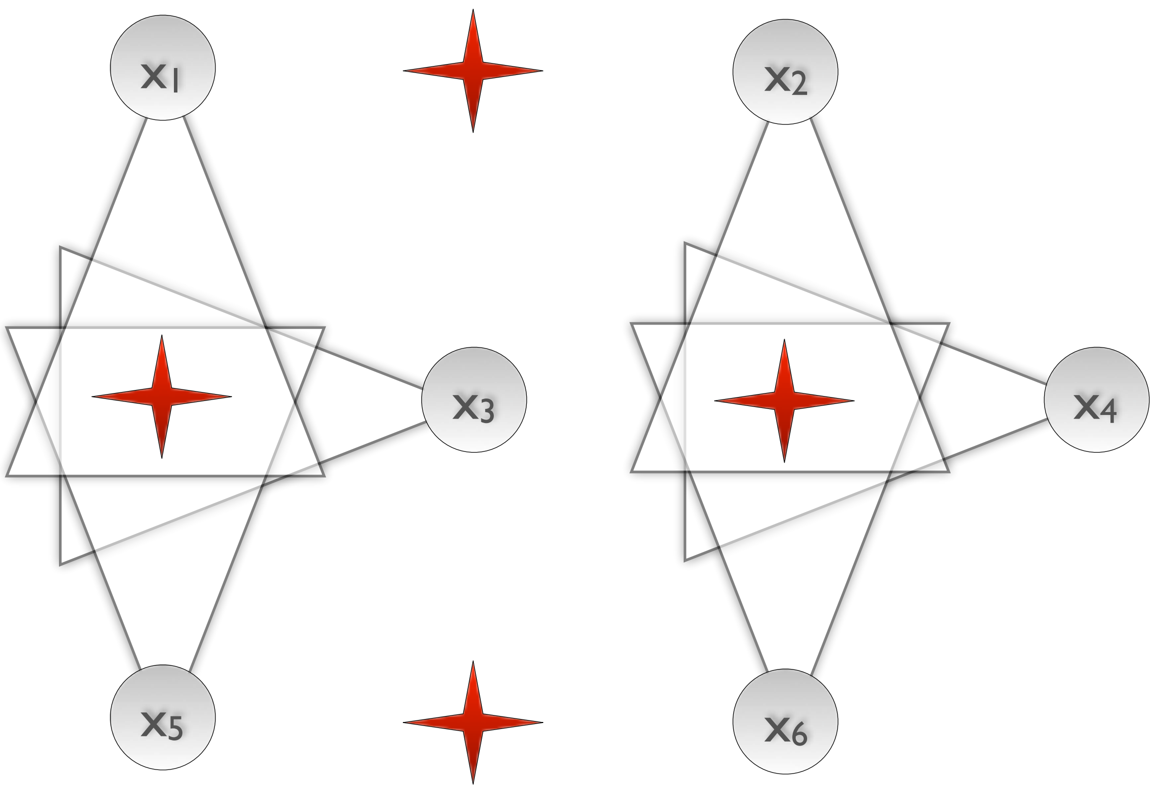


x_1	x_3	x_5	Sat?
N	*N*	*N*	X
N	*N*	*E*	X
	…		X
S	***W***	***N***	✓
	…		X
W	*W*	*W*	X

Model the problem as a CSP

Max-CSP

MAX CONSTRAINT SATISFACTION



x_1	x_3	x_5	Sat?
N	*N*	*N*	X
N	*N*	*E*	X
…			X
S	***W***	***N***	✓
…			X
W	*W*	*W*	X

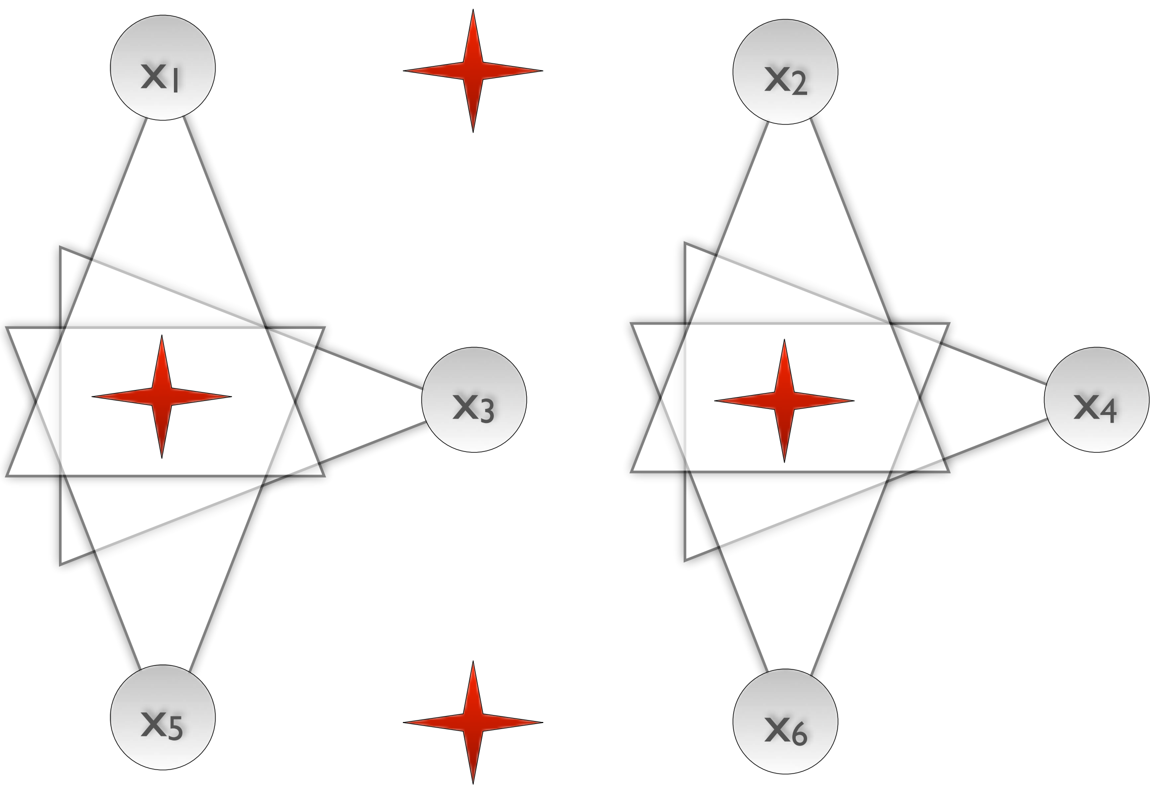
Model the problem as a Max-CSP

Max-CSP
MAX CONSTRAINT SATISFACTION

- Variables $X = \{x_1, \ldots, x_n\}$
- Domains $D = \{D_1, \ldots, D_n\}$
- Constraints $C = \{c_1, \ldots, c_m\}$
  where a constraint $c_i \subseteq D_{i_1} \times D_{i_2} \times \ldots \times D_{i_n}$ denotes the possible valid joint assignments for the variables $x_{i_1}, x_{i_2}, \ldots, x_{i_n}$ it involves

- **GOAL**: Find an assignment to all variables that satisfies a maximum number of constraints

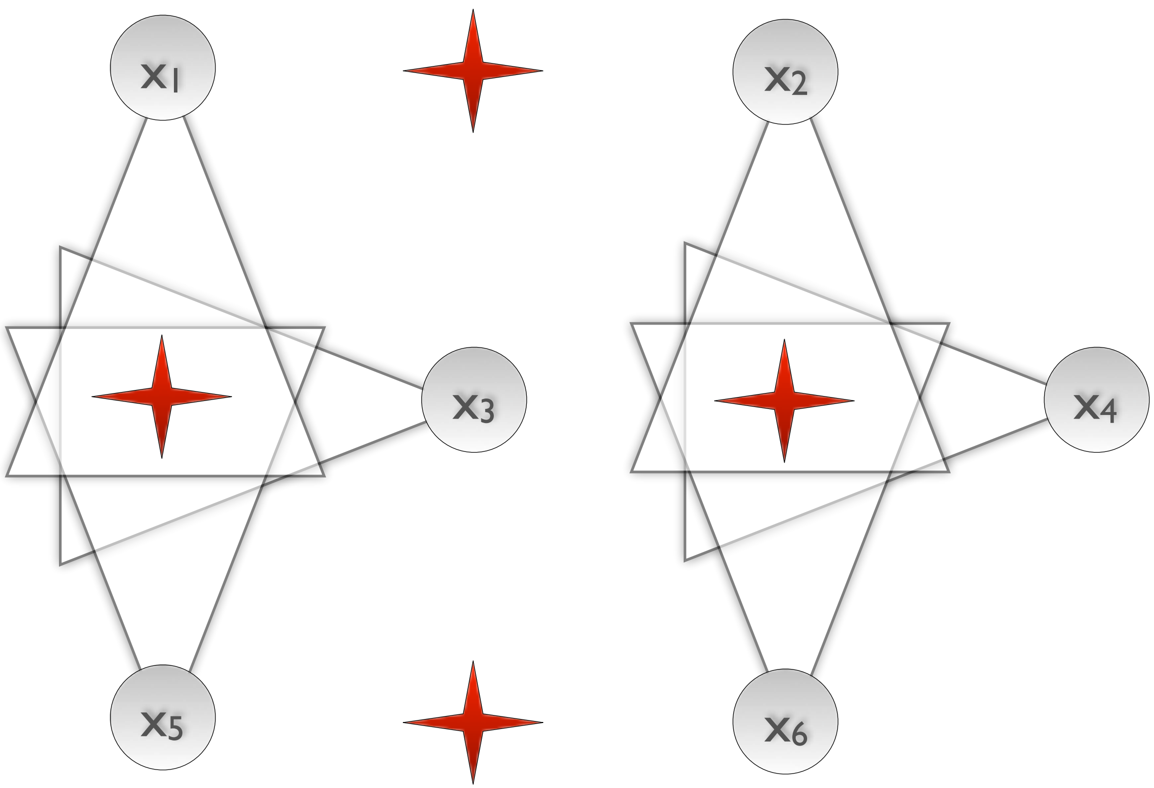
Max-CSP
MAX CONSTRAINT SATISFACTION



x_1	x_3	x_5	Sat?
N	*N*	*N*	X
N	*N*	*E*	X
…			X
S	***W***	***N***	✓
…			X
W	*W*	*W*	X

Model the problem as a Max-CSP

WCSP (COP)
CONSTRAINT OPTIMIZATION



x_1	x_3	x_5	Cost
N	*N*	*N*	∞
N	*N*	*E*	∞
	...		∞
S	***W***	***N***	**10**
	...		∞
W	*W*	*W*	∞

Model the problem as a COP

WCSP (COP)

CONSTRAINT OPTIMIZATION

- Variables $X = \{x_1, \ldots, x_n\}$
- Domains $D = \{D_1, \ldots, D_n\}$
- Constraints $C = \{c_1, \ldots, c_m\}$
  where a constraint $c_i : D_{i_1} \times \ldots \times D_{i_n} \rightarrow \mathbb{R}_+ \cup \{\infty\}$
  expresses the degree of constraint violation

- **GOAL**: Find an assignment that minimizes the sum of the costs of all the constraints

WCSP (COP)
CONSTRAINT OPTIMIZATION

- Objective: maximize #constraints satisfied

WCSP (COP)
CONSTRAINT OPTIMIZATION

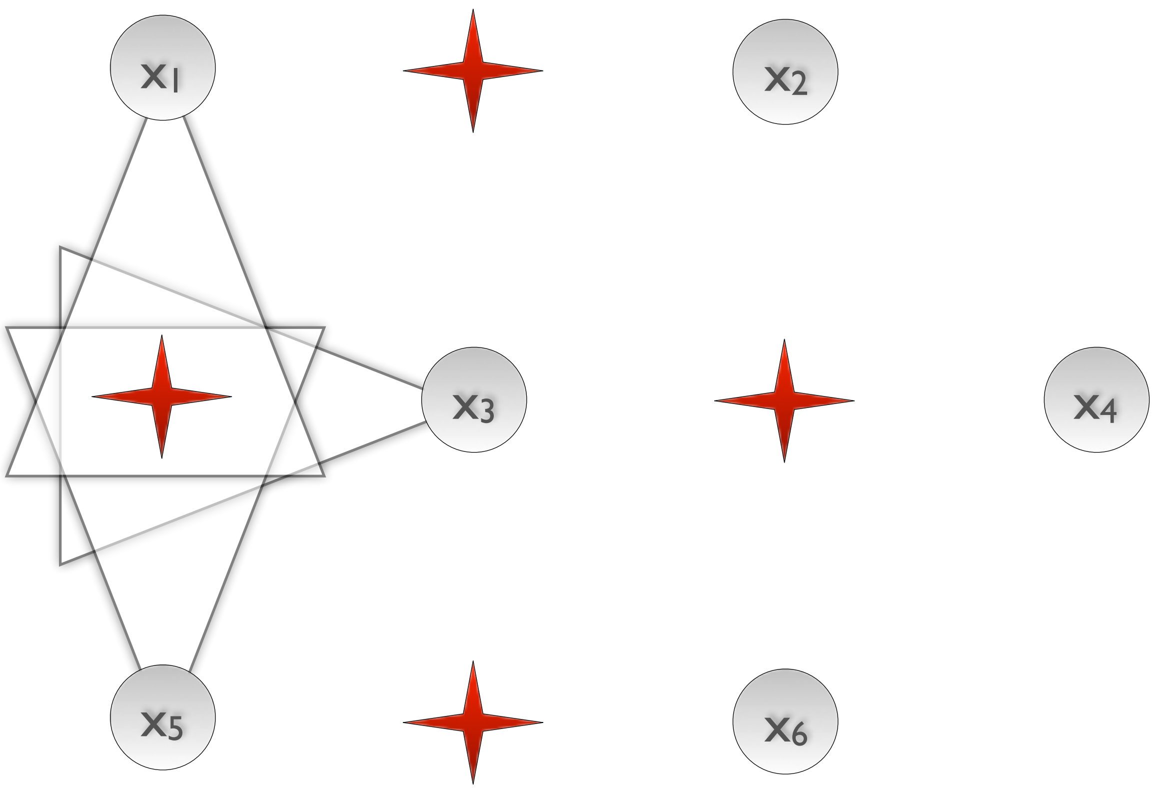
CSP → **Max-CSP**

- Objective: maximize #constraints satisfied

CSP → **COP**; **Max-CSP** → **COP**

- Hard constraints to Soft constraints
- Objective: minimize cost

WCSP (COP)
CONSTRAINT OPTIMIZATION



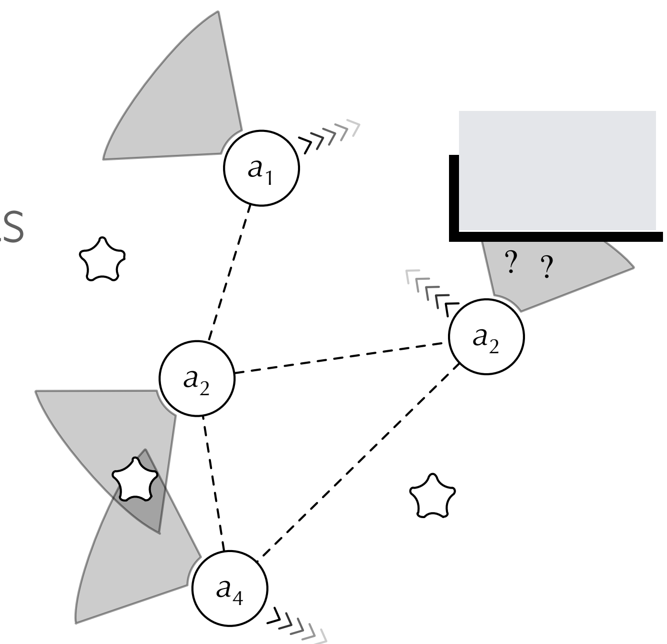
Imagine that each sensor is an autonomous agent.

How should this problem be modeled and solved in a decentralized manner?

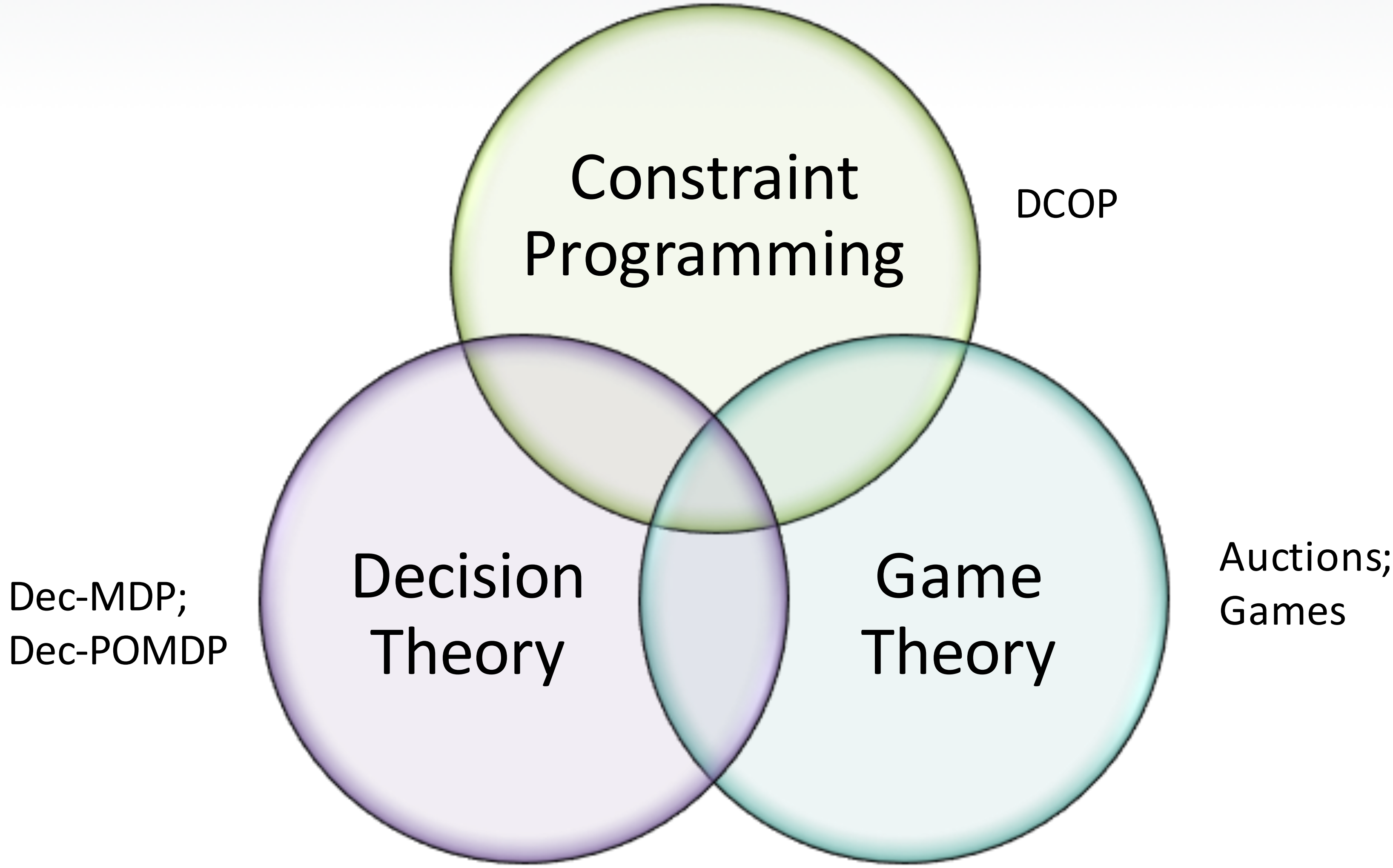
Multi-Agent Systems

- **Agent**: An entity that behaves autonomously in the pursuit of goals
- **Multi-agent system**: A system of multiple interacting agents

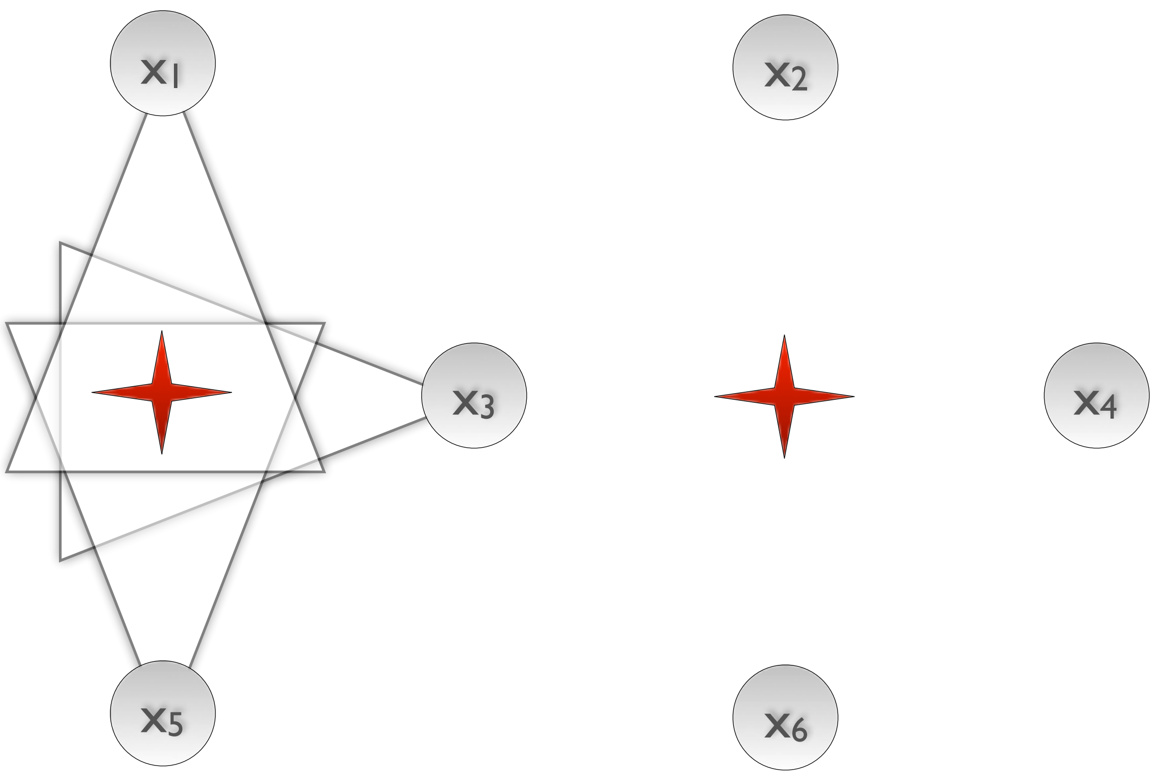
- An agent is:
  - *Autonomous*: Is of full control of itself
  - *Interactive*: May communicate with other agents
  - *Reactive*: Responds to changes in the environment or requests by other agents
  - *Proactive*: Takes initiatives to achieve its goals



Multi-Agent Systems



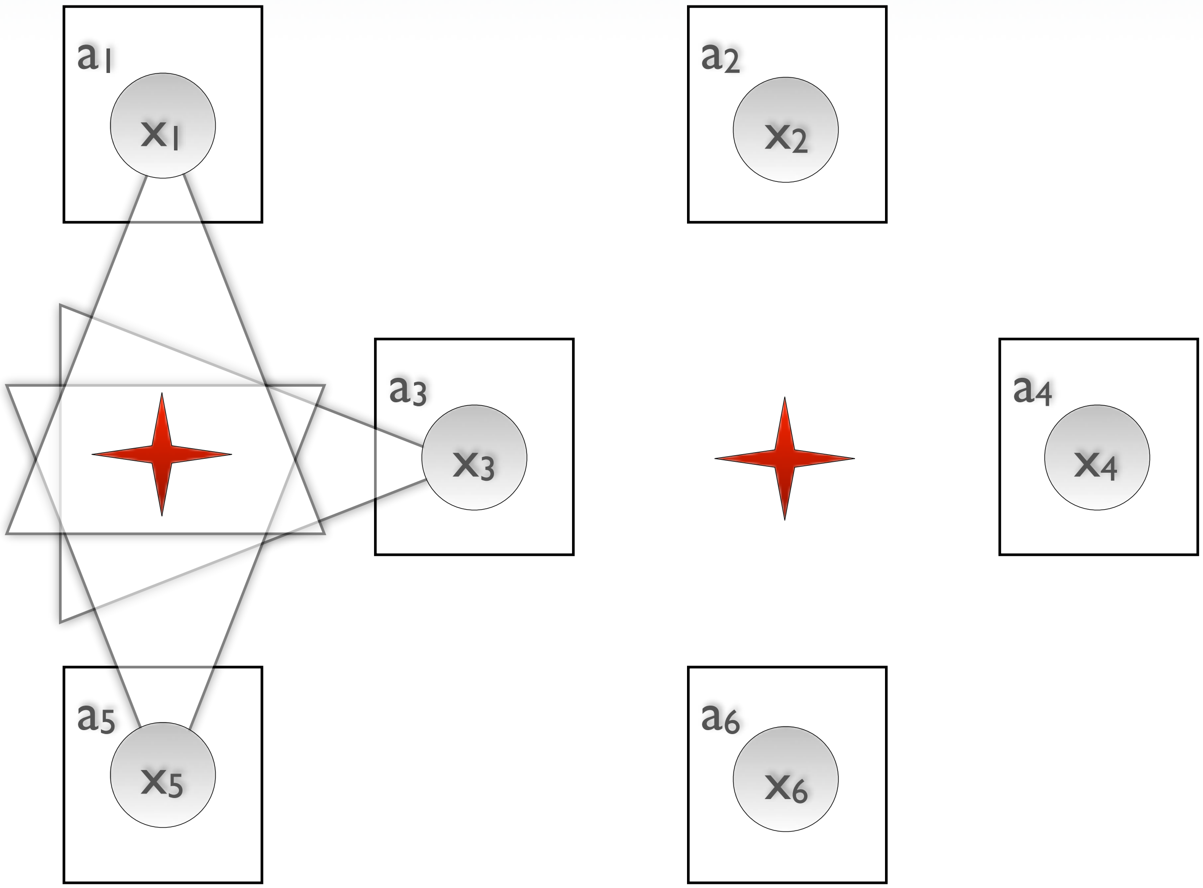
DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION



Imagine that each sensor is an autonomous agent.

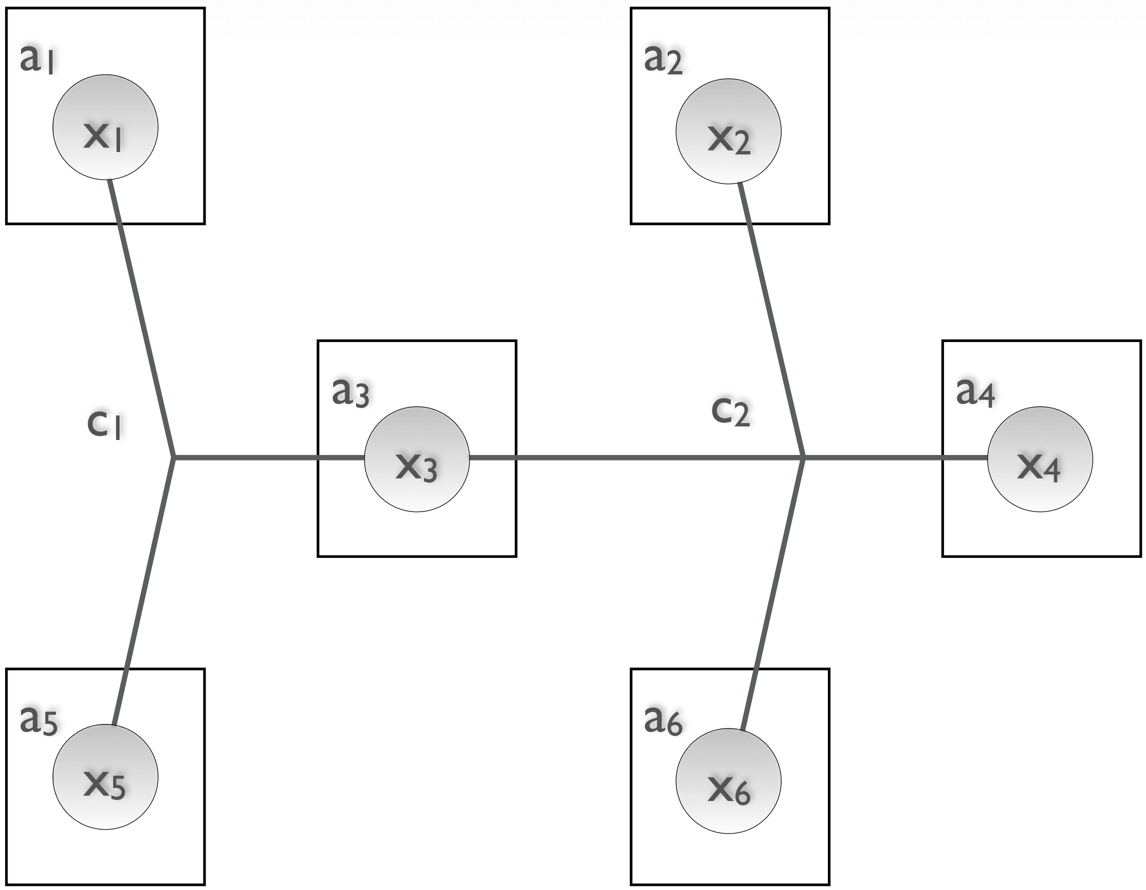
How should this problem be modeled and solved in a decentralized manner?

DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION



DCOP

DISTRIBUTED CONSTRAINT OPTIMIZATION



DCOP

DISTRIBUTED CONSTRAINT OPTIMIZATION

- Agents $A = \{a_i, \ldots, a_n\}$
- Variables $X = \{x_1, \ldots, x_n\}$
- Domains $D = \{D_1, \ldots, D_n\}$
- Constraints $C = \{c_1, \ldots, c_m\}$
- Mapping of variables to agents

- **GOAL**: Find an assignment that minimizes the sum of the costs of all the constraints

DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION

CSP

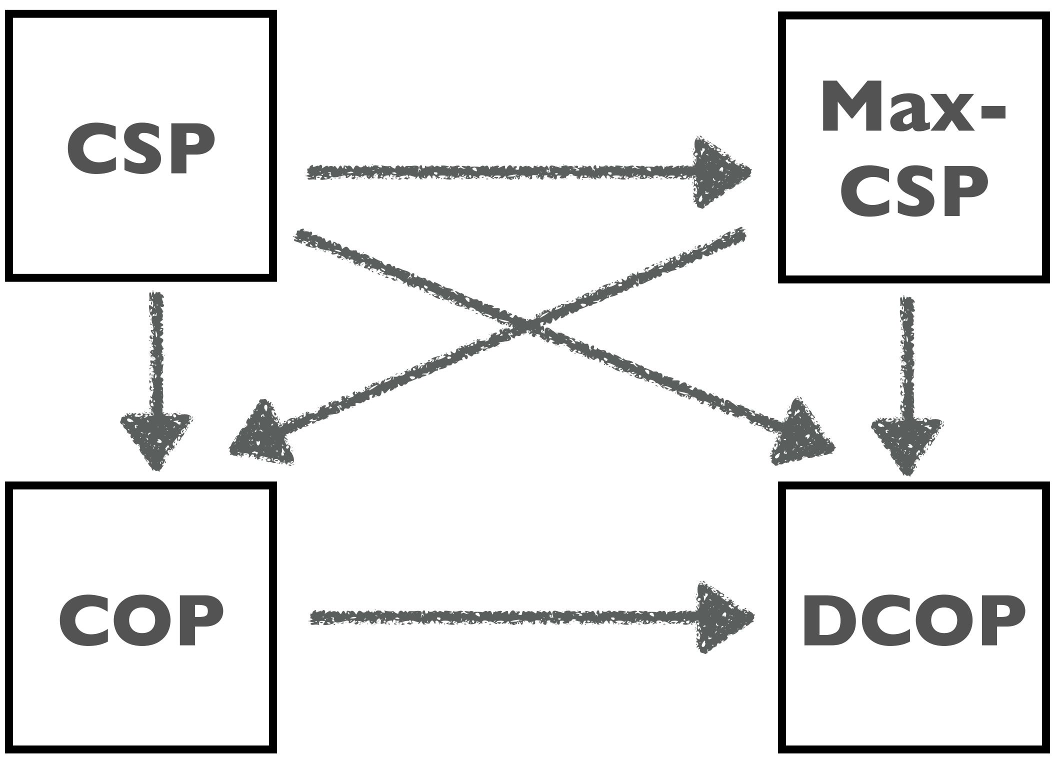
Max-CSP

- Objective: maximize #constraints satisfied

COP

- Hard constraints to Soft constraints
- Objective: minimize cost

DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION



- Variables are controlled by agents
- Communication model
- Local agents' knowledge

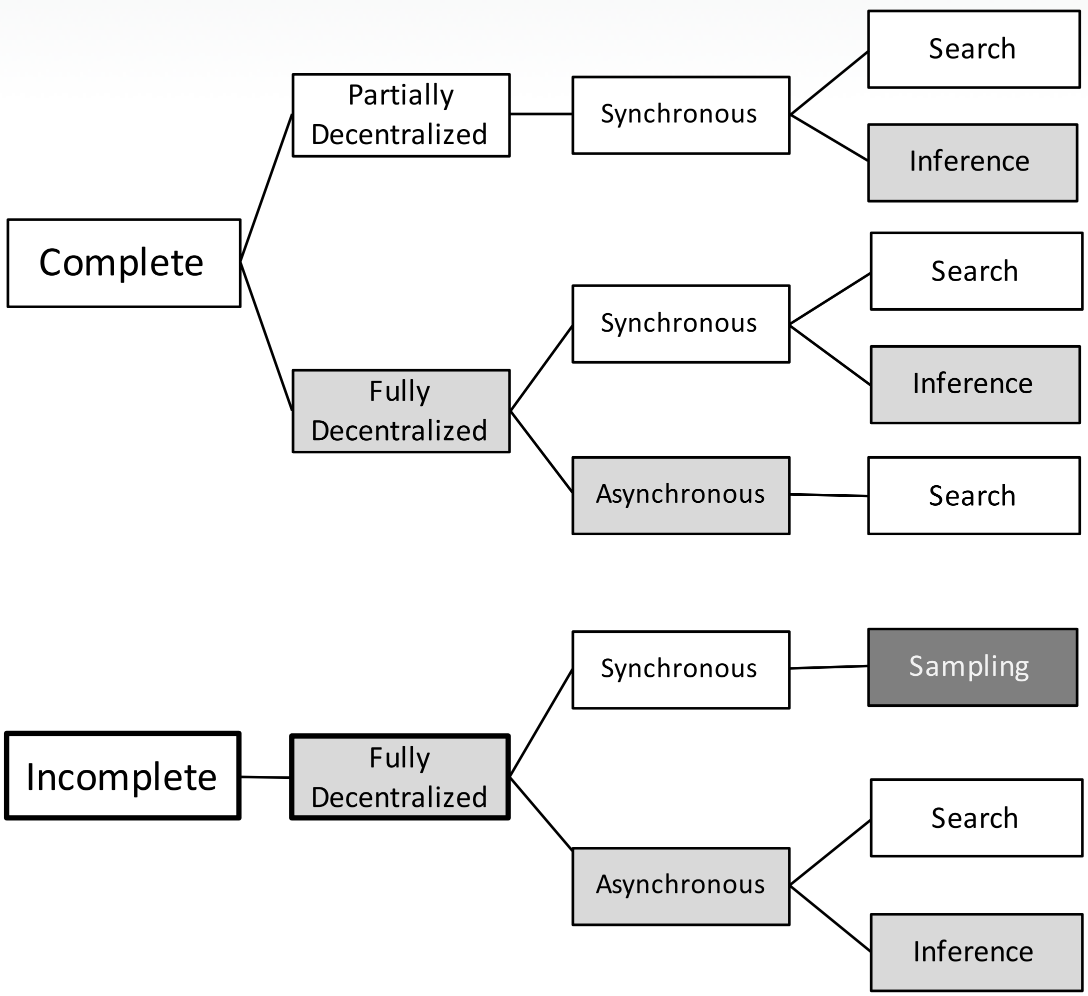
DCOP
DISTRIBUTED CONSTRAINT OPTIMIZATION

- Why distributed models?
  - Natural mapping for multi-agent systems
  - Potentially faster by exploiting parallelism
  - Potentially more robust: no single point of failure, no single network bottleneck
  - Maintains more private information
  - …

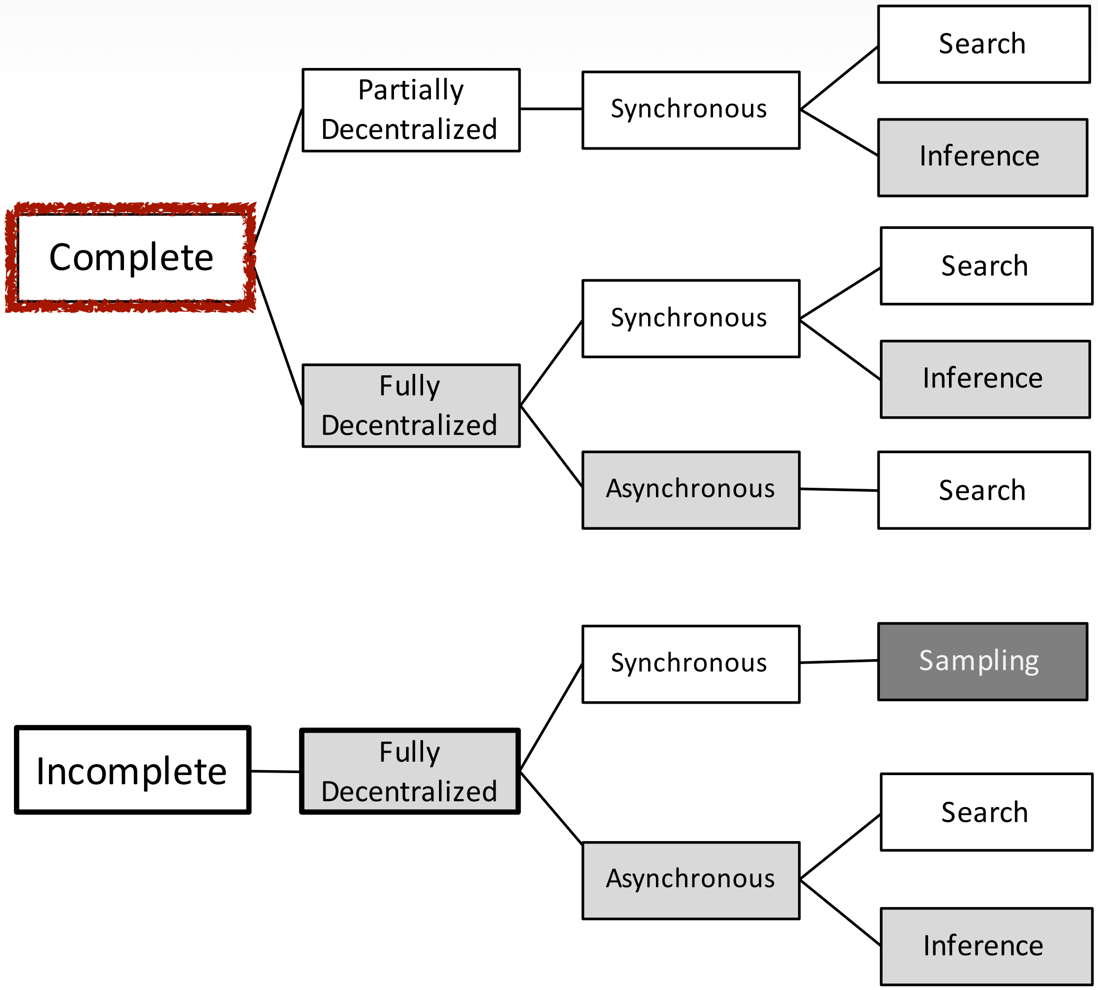
DCOP Algorithms

AAAI-20 Tutorial on
Multi-Agent Distributed Constrained Optimization

DCOP Algorithms

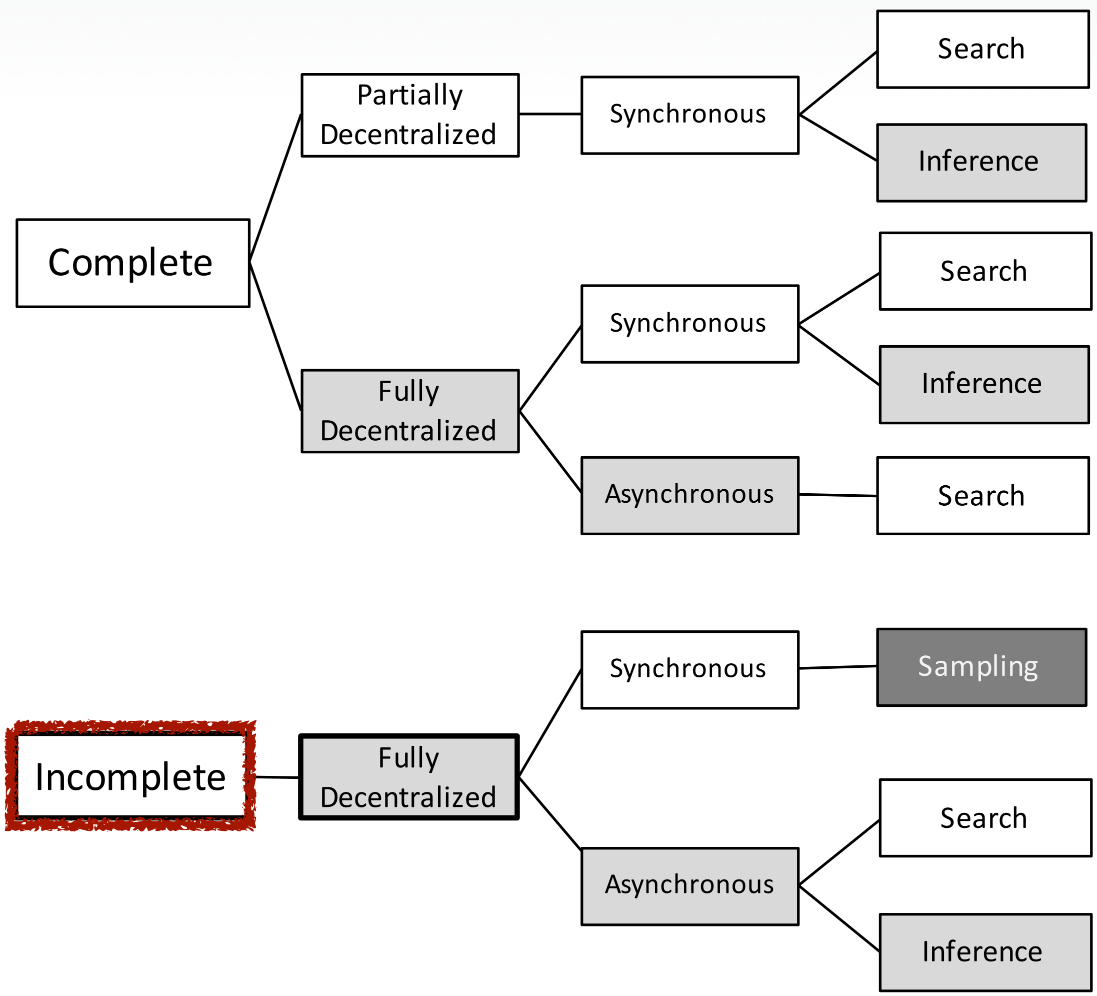


DCOP Algorithms



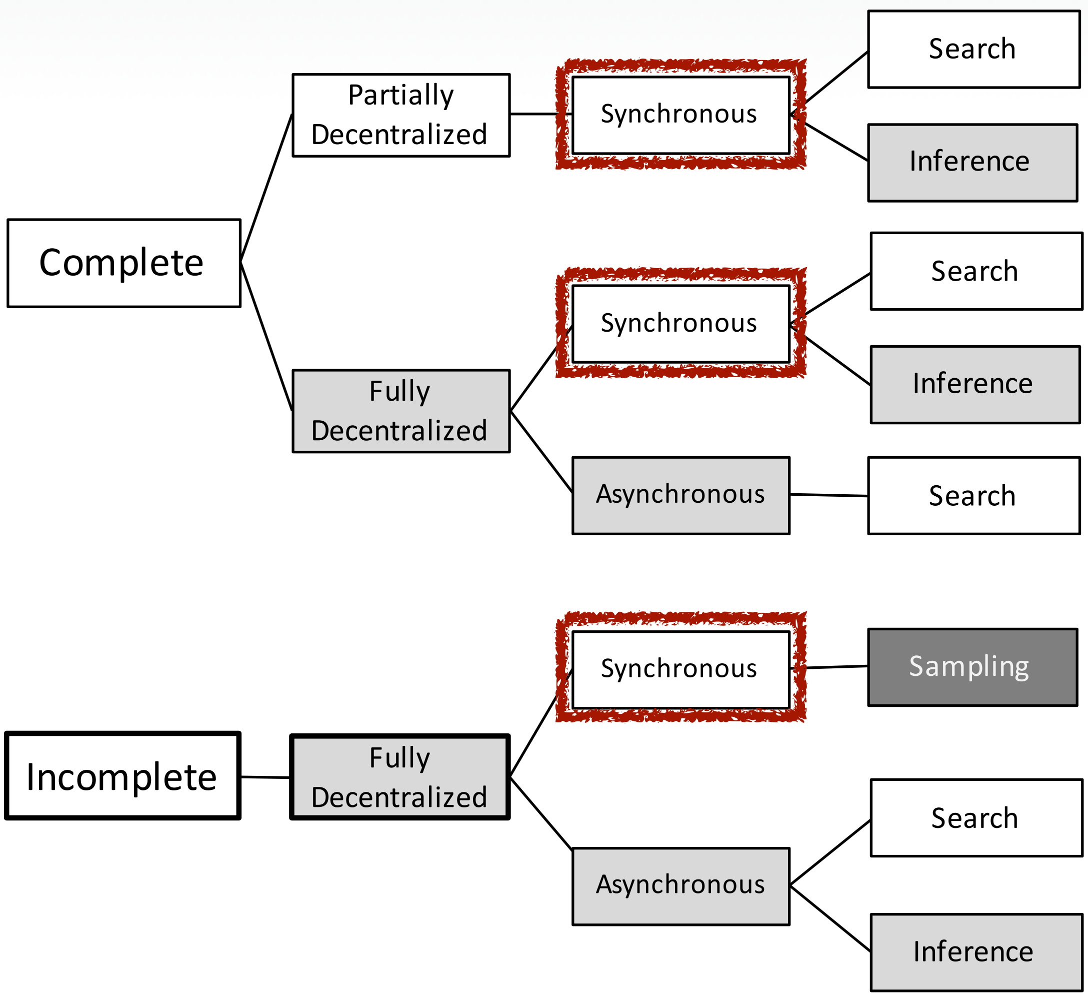
- Important Metrics:
  - Agent complexity
  - Network loads
  - Message size

DCOP Algorithms



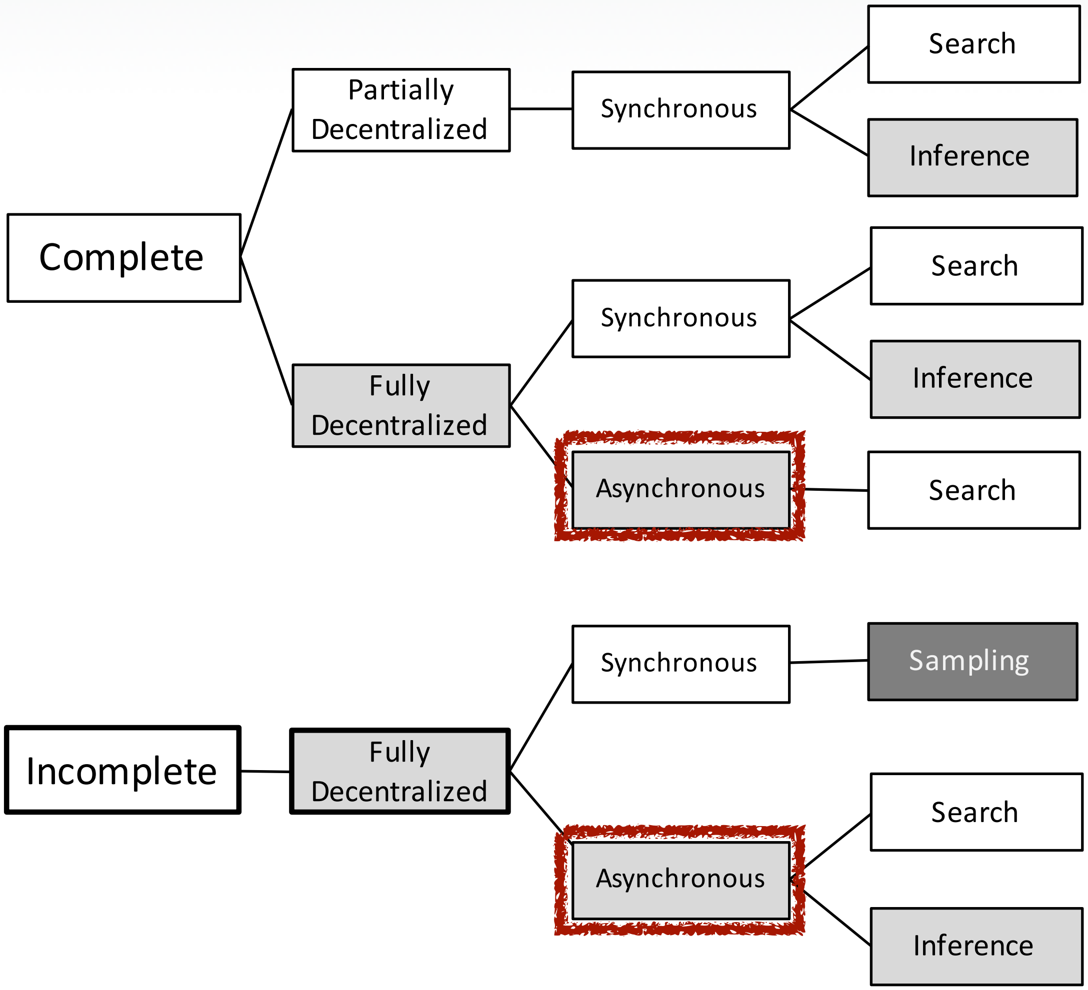
- Important Metrics:
  - Agent complexity
  - Network loads
  - Message size
  - Anytime
  - Quality guarantees
  - Execution time vs. solution quality

DCOP Algorithms



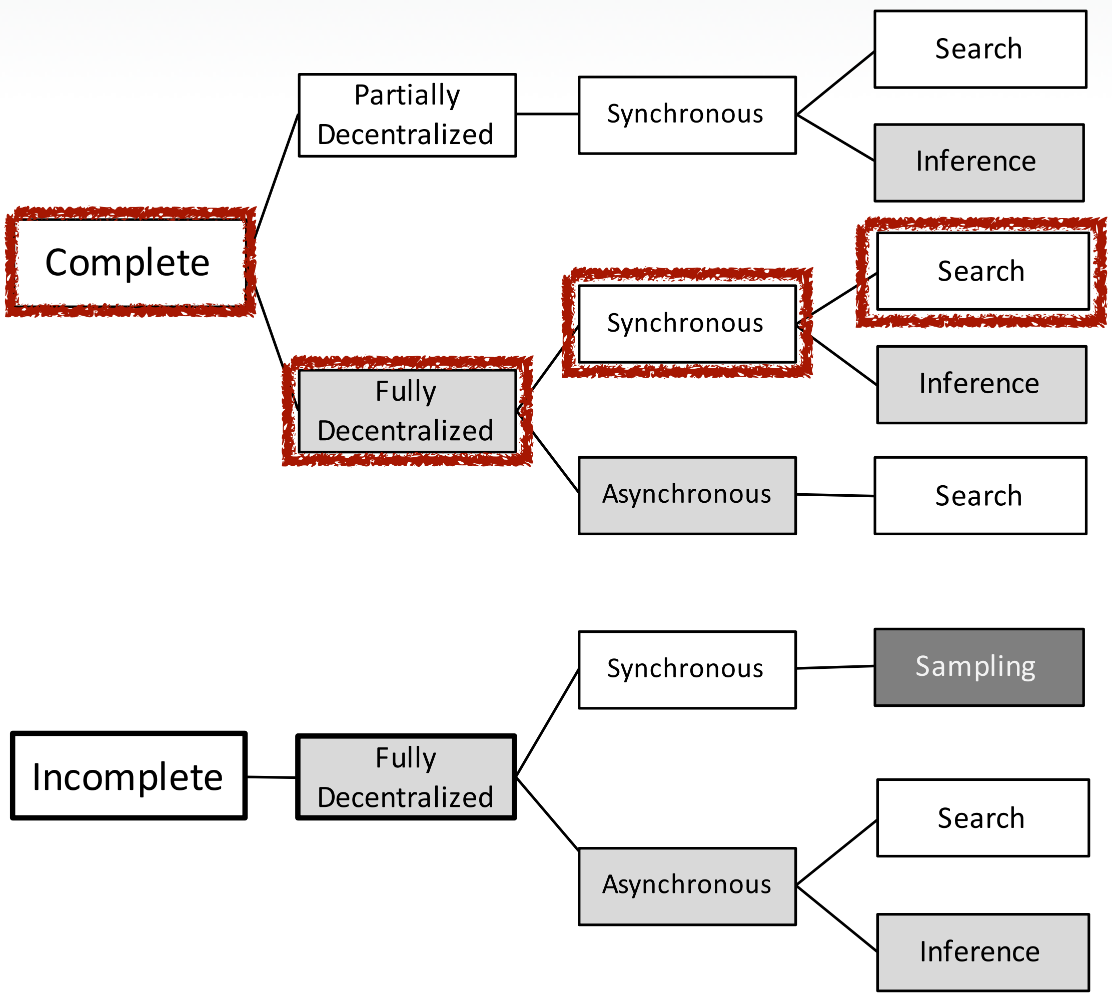
- Systematic process, divided in steps.
- Each agent waits for particular messages before acting
- Consistent view of the search process
- Typically, increases idle-time

DCOP Algorithms



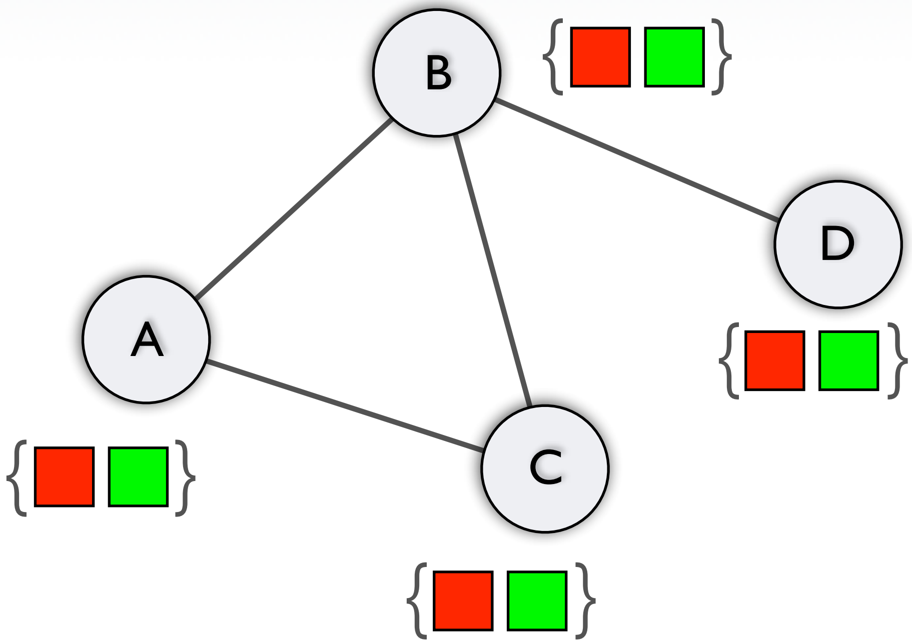
- Decision based on agents' local state
- Agents' actions do not depend on sequence of received messages
- Minimizes idle-time
- No guarantees on validity of local views

DCOP Algorithms



Synchronous Branch and Bound (SBB)

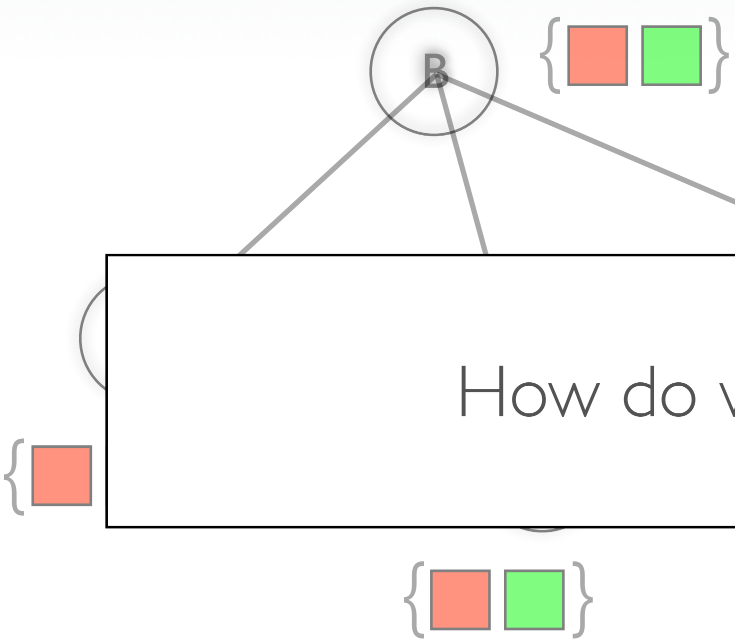
SBB



x_i	x_j	Cost (A,B)	Cost (A,C)	Cost (B,C)	Cost (B,D)
		5	5	5	3
		8	10	4	8
		20	20	3	10
		3	3	3	3

Katsutoshi Hirayama, Makoto Yokoo: Distributed Partial Constraint Satisfaction Problem. CP 1997: 222-236

SBB



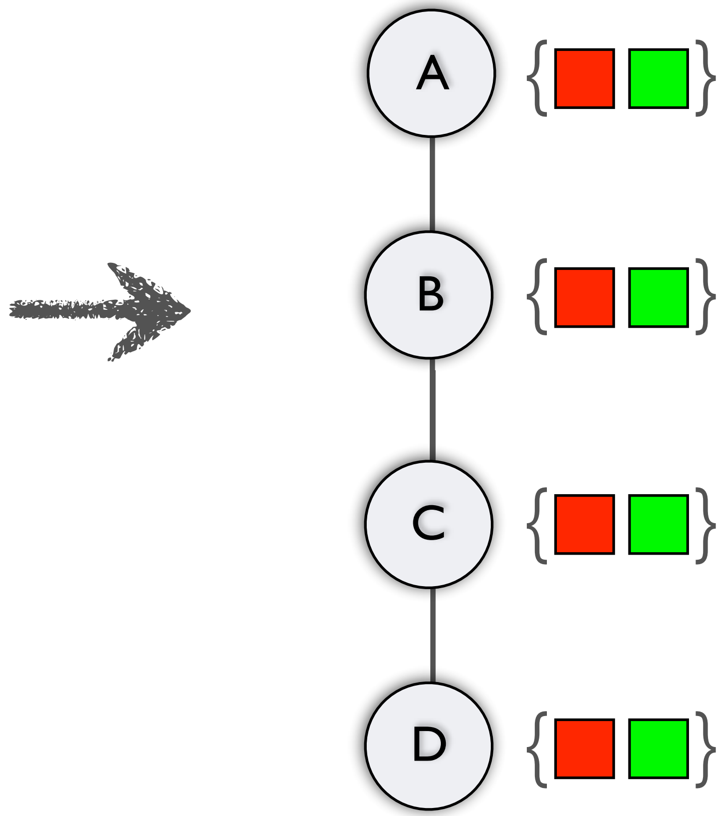
x_i	x_j	Cost (A,B)	Cost (A,C)	Cost (B,C)	Cost (B,D)
		5	5	5	3
					8
					10
		3	3	3	3

How do we solve this distributedly?

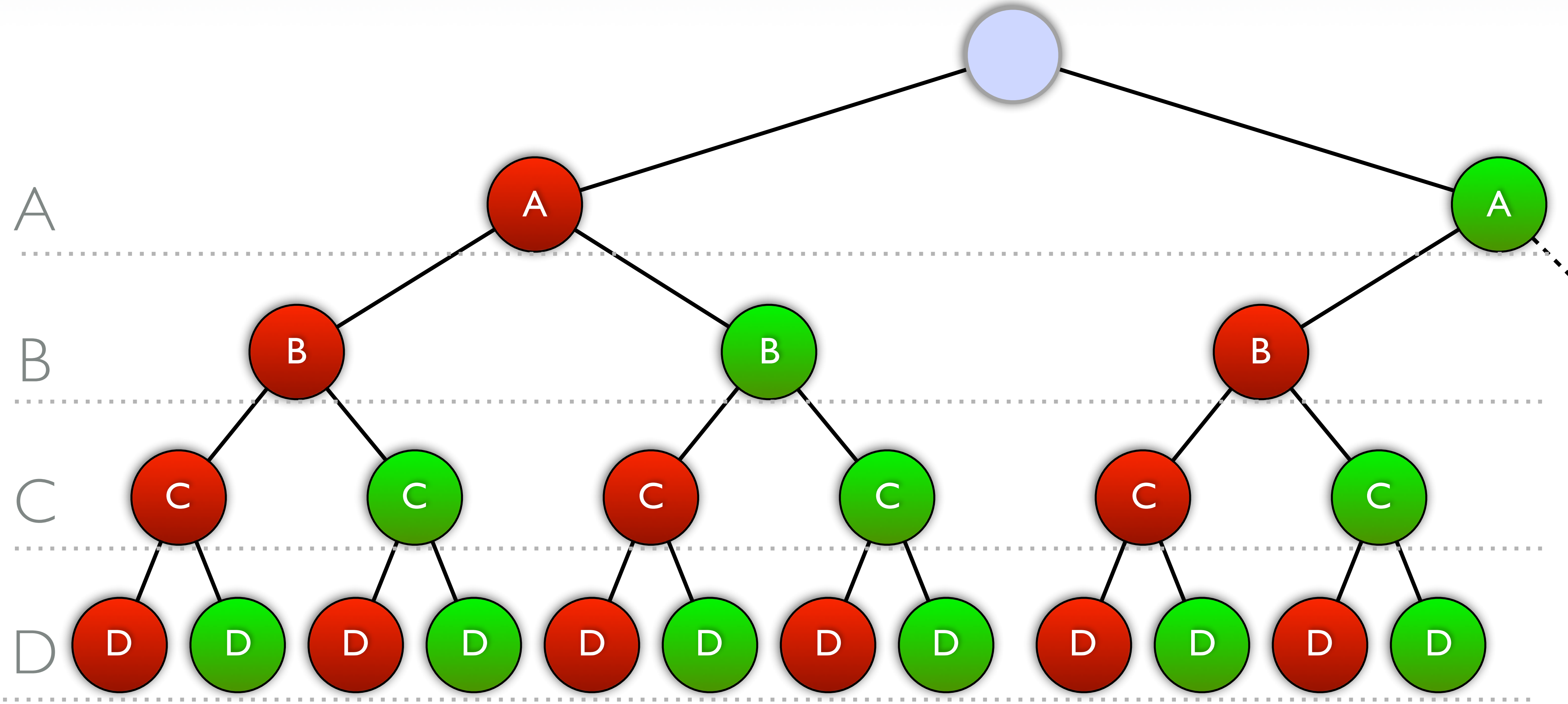
SBB

- Agents operate on a complete ordering
- Agents exchange CPA messages containing partial assignments.
- When a solution is found, its solution cost as an UB is broadcasted to all agents.
- The UB is used for branch pruning.

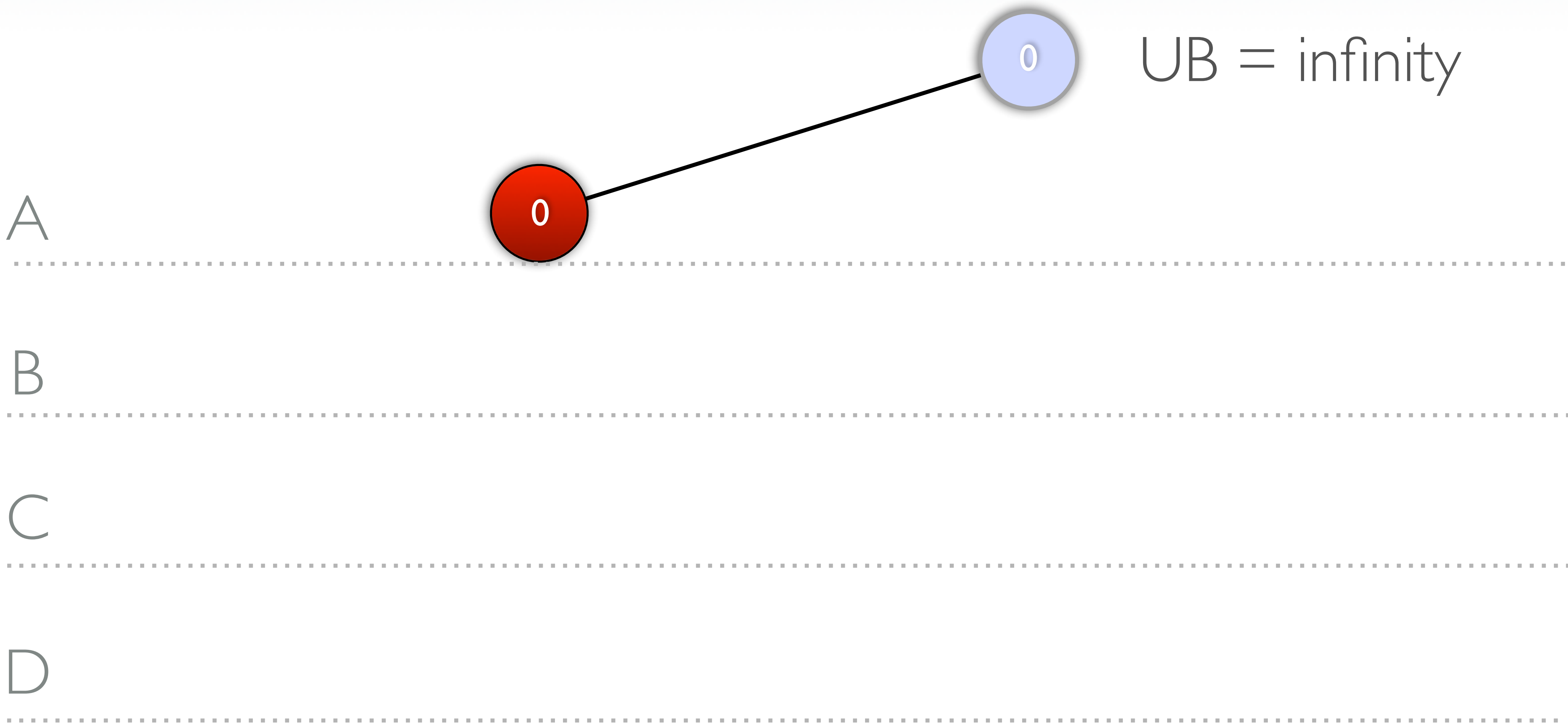
Complete Ordering



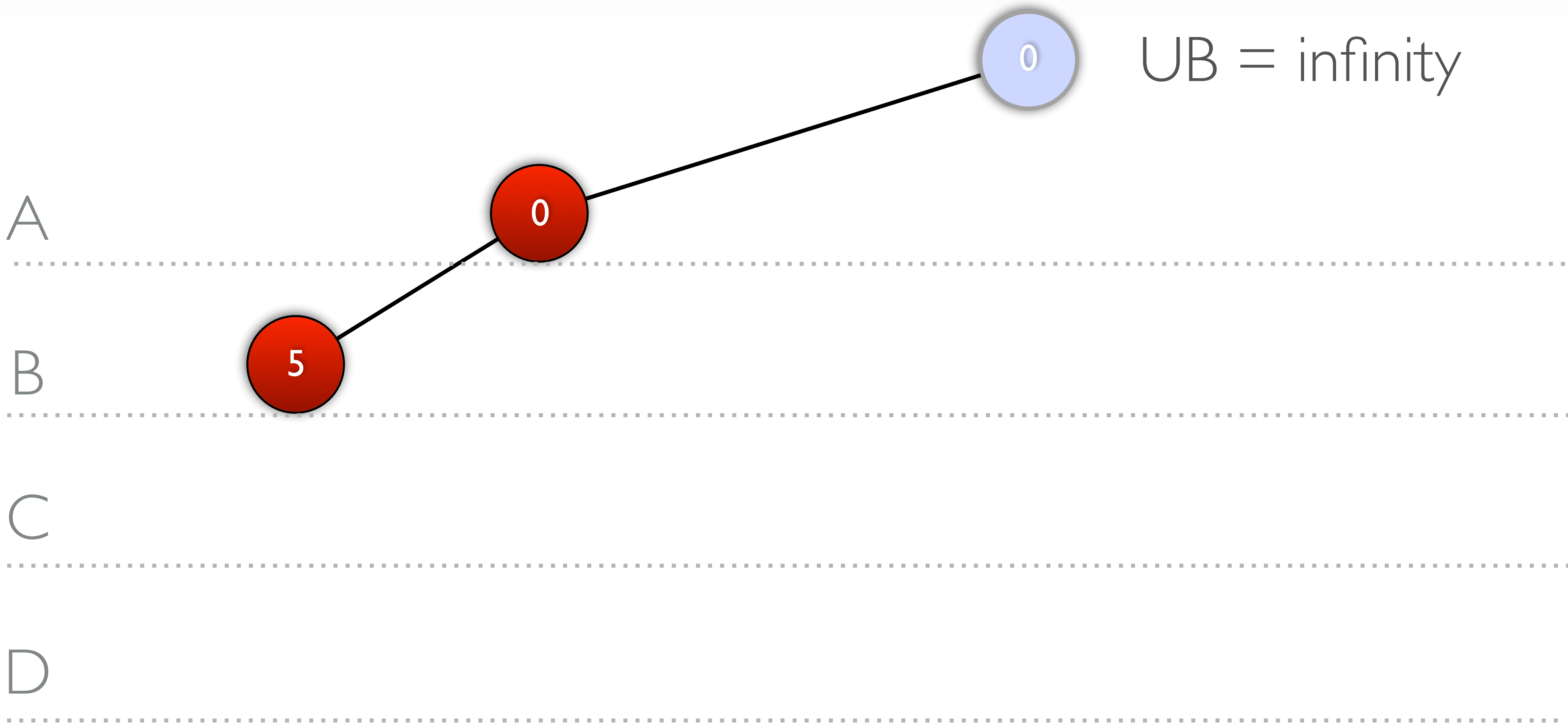
SBB



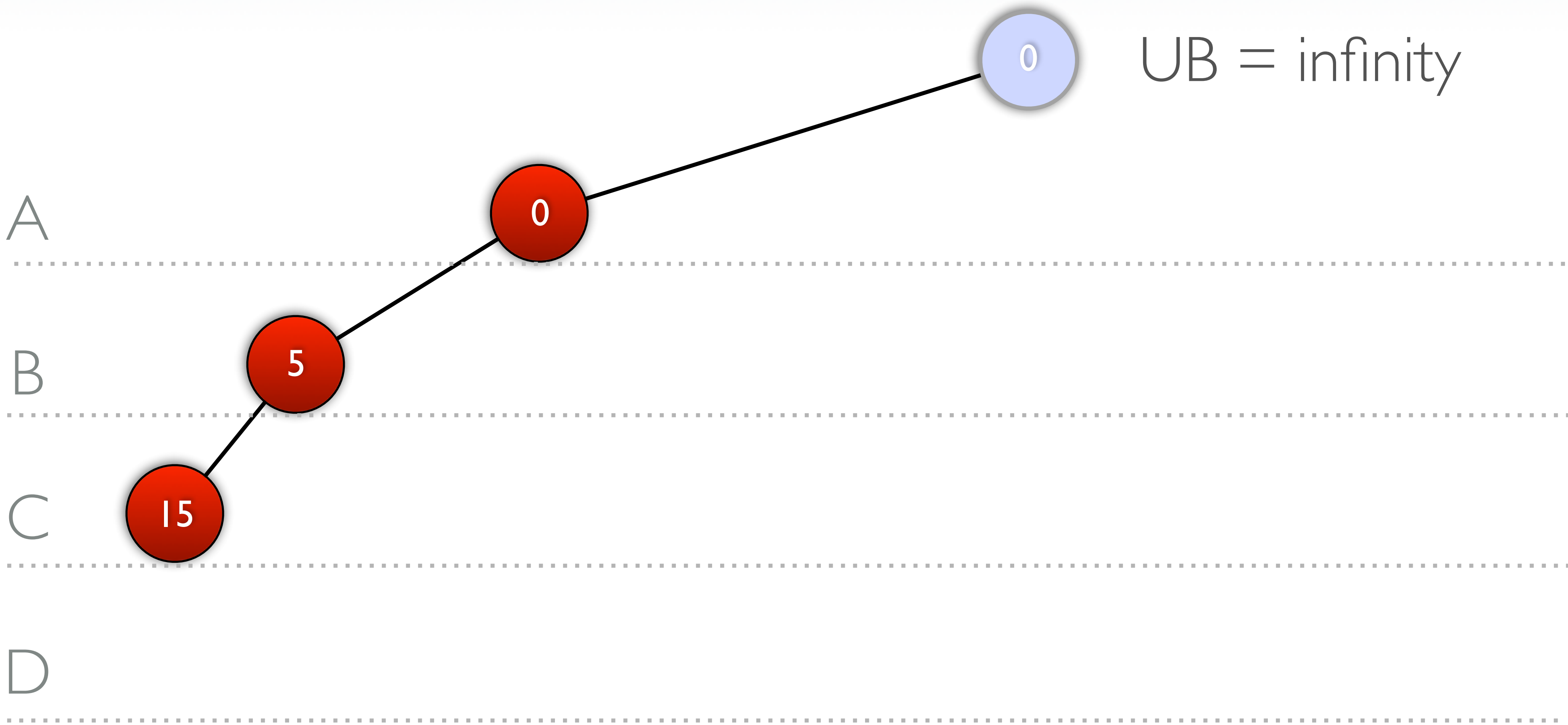
SBB



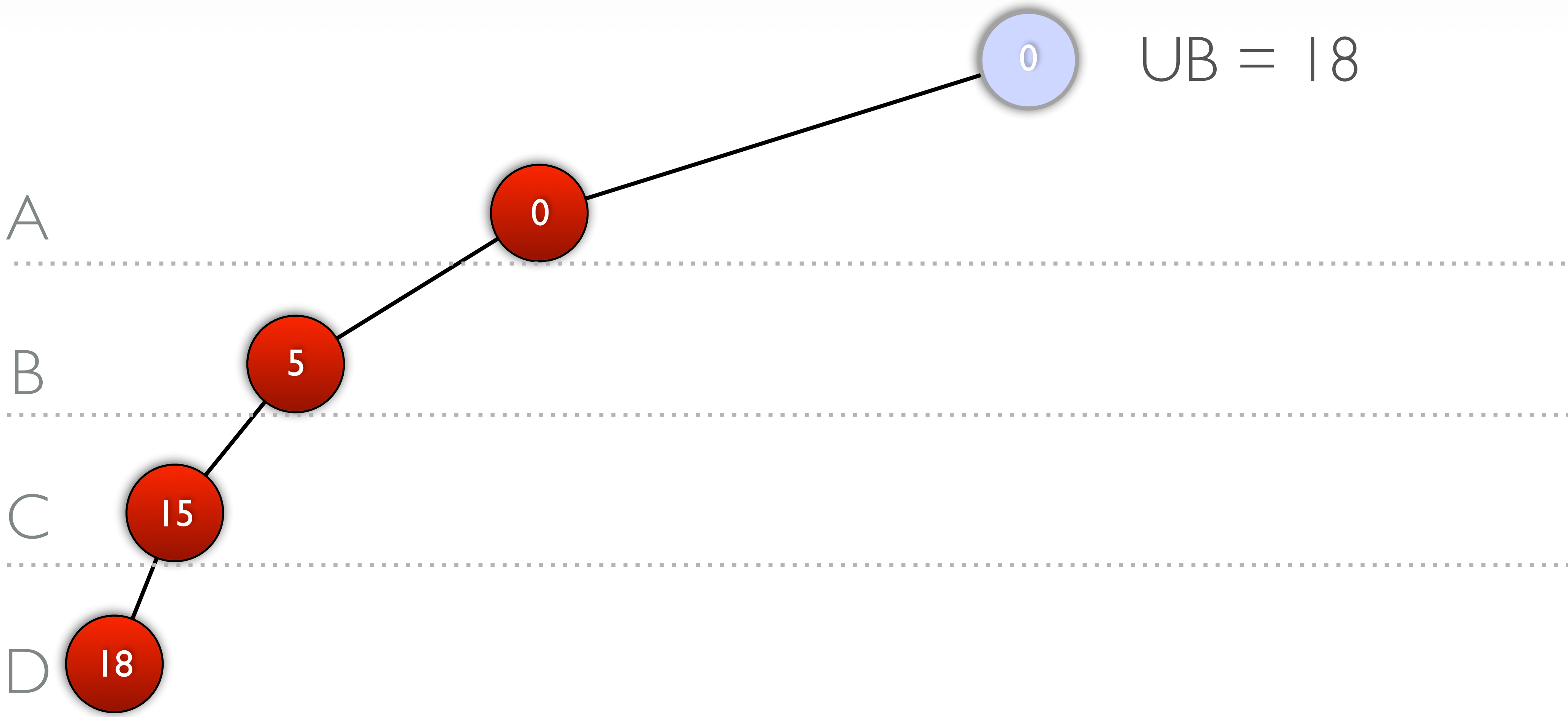
SBB



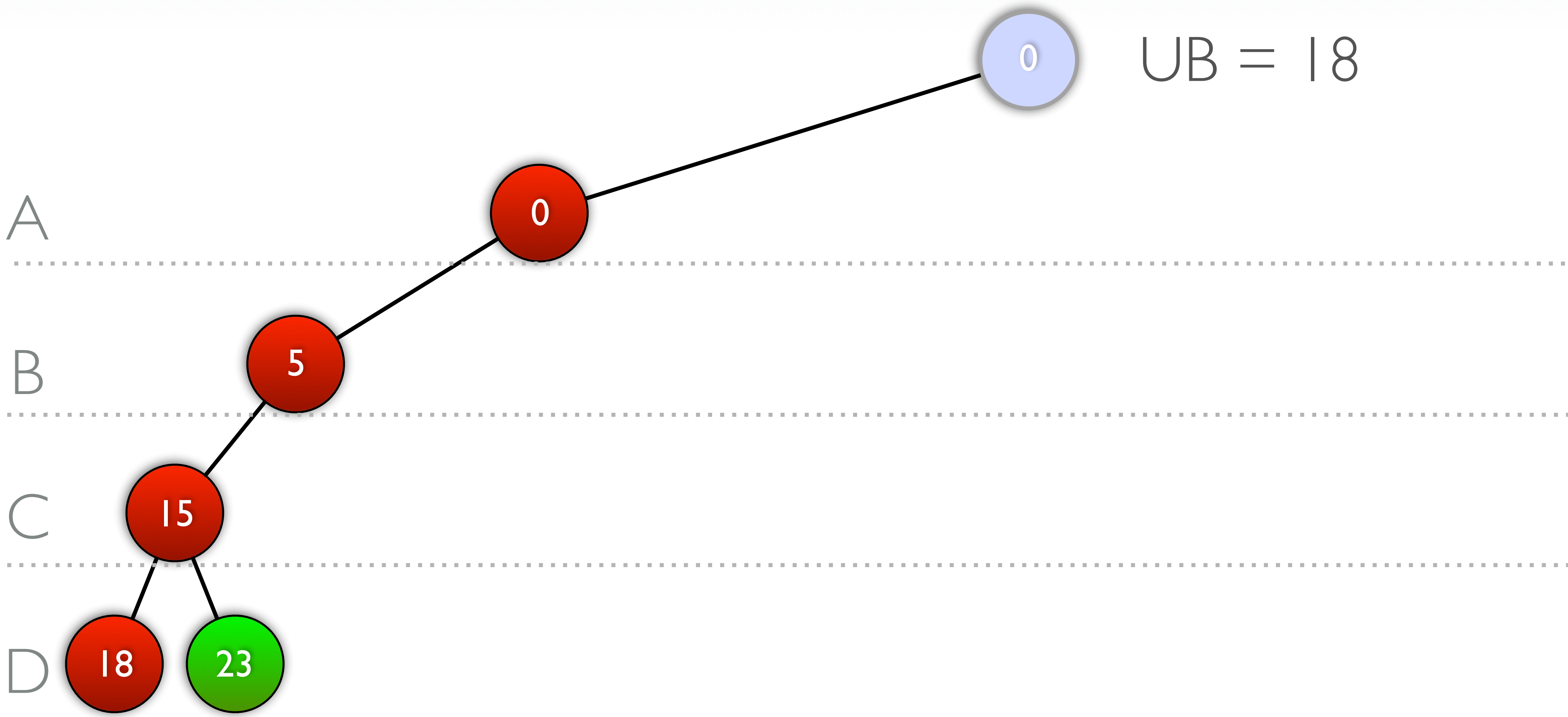
SBB



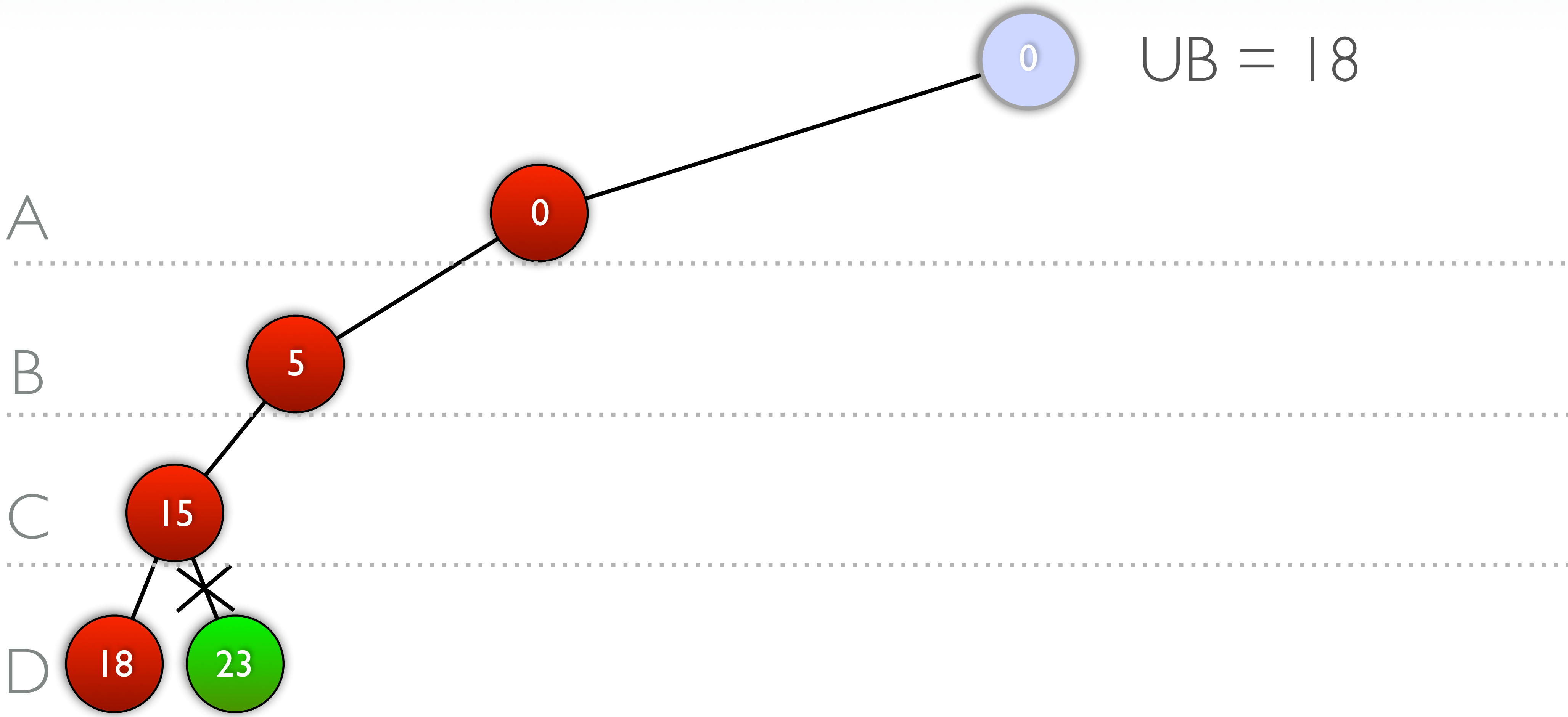
SBB



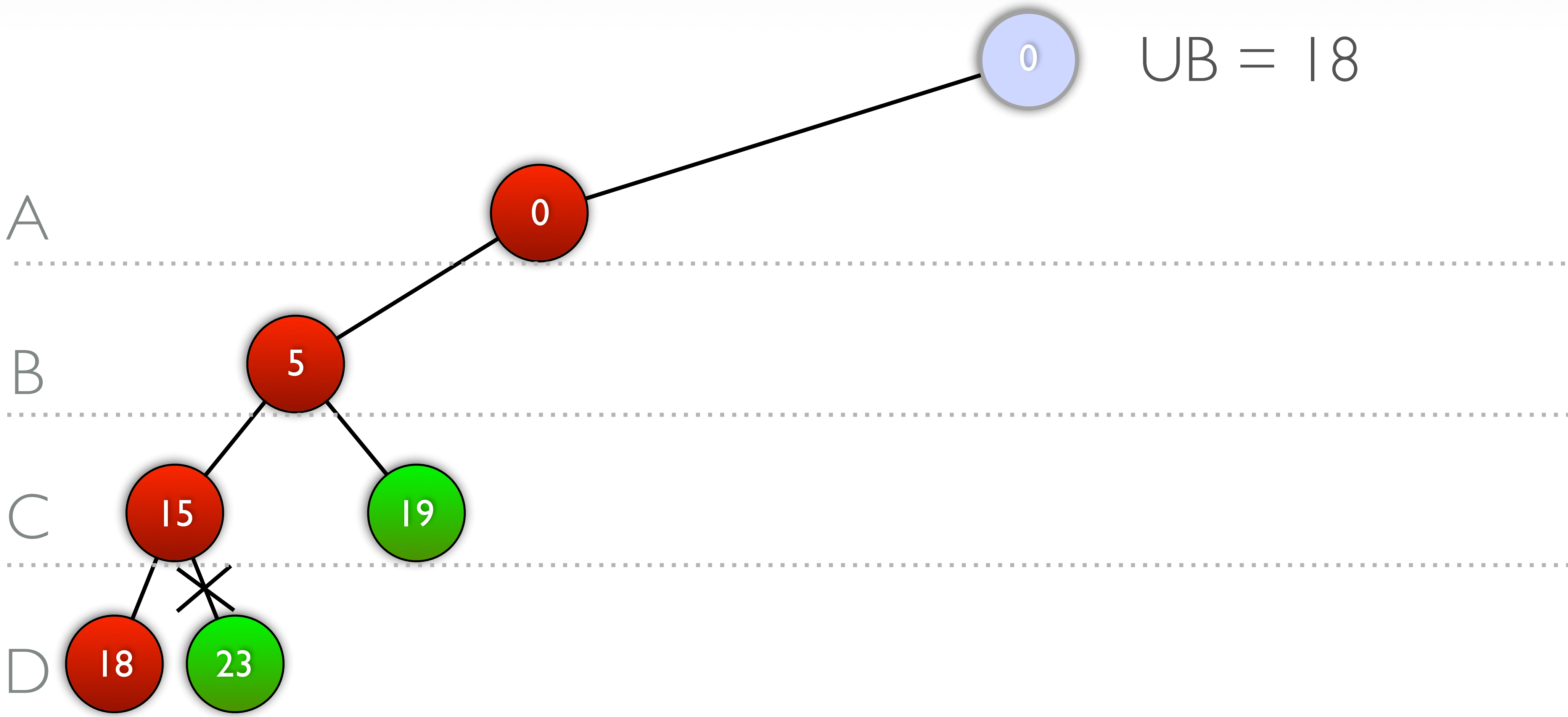
SBB



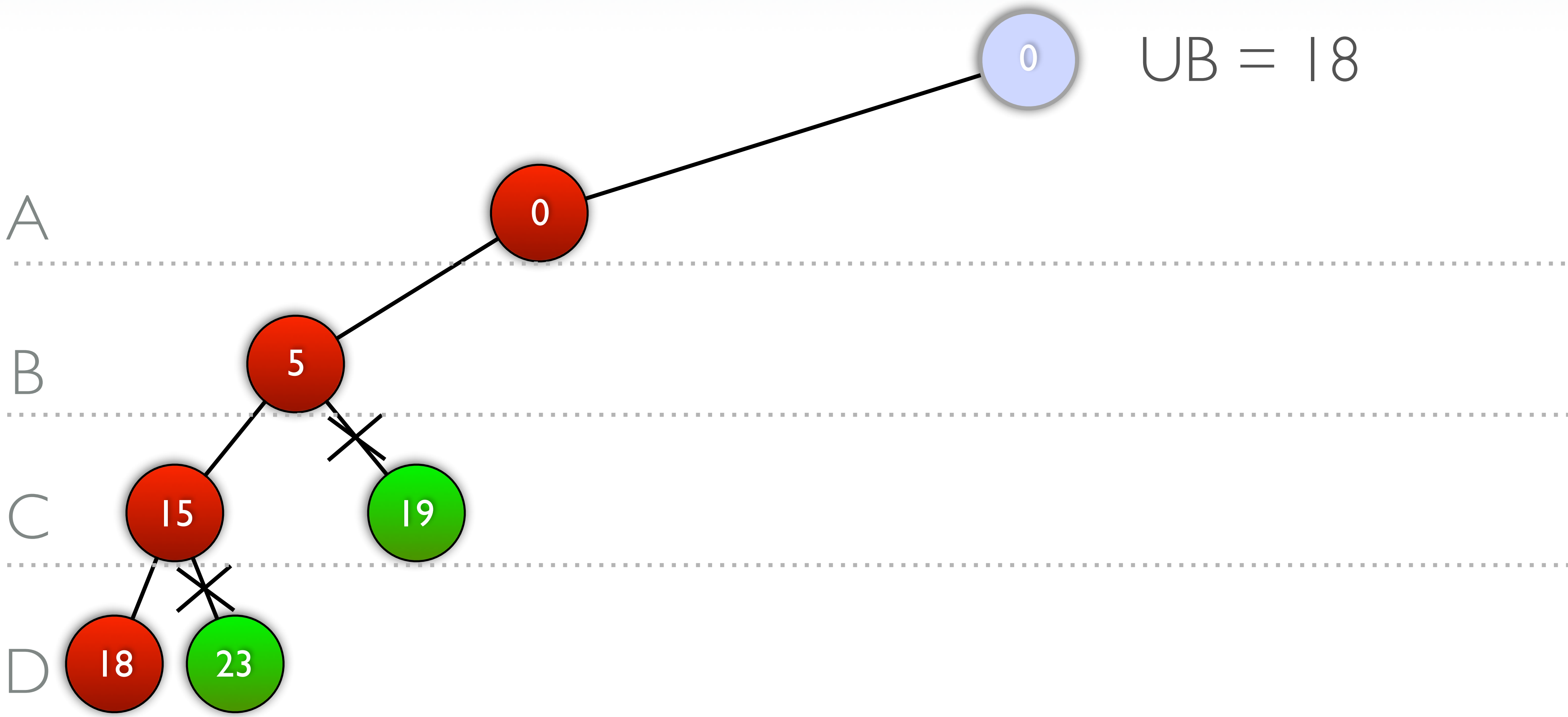
SBB



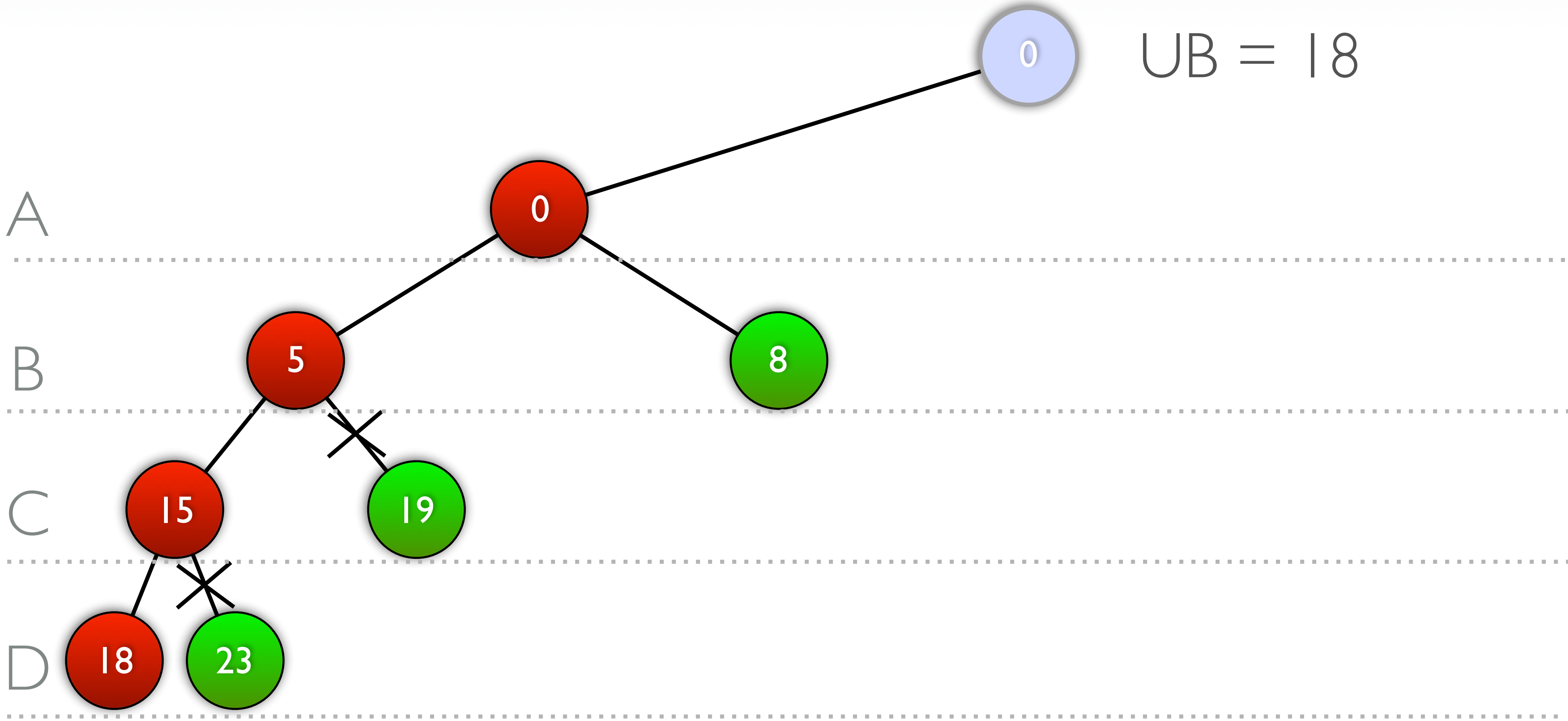
SBB



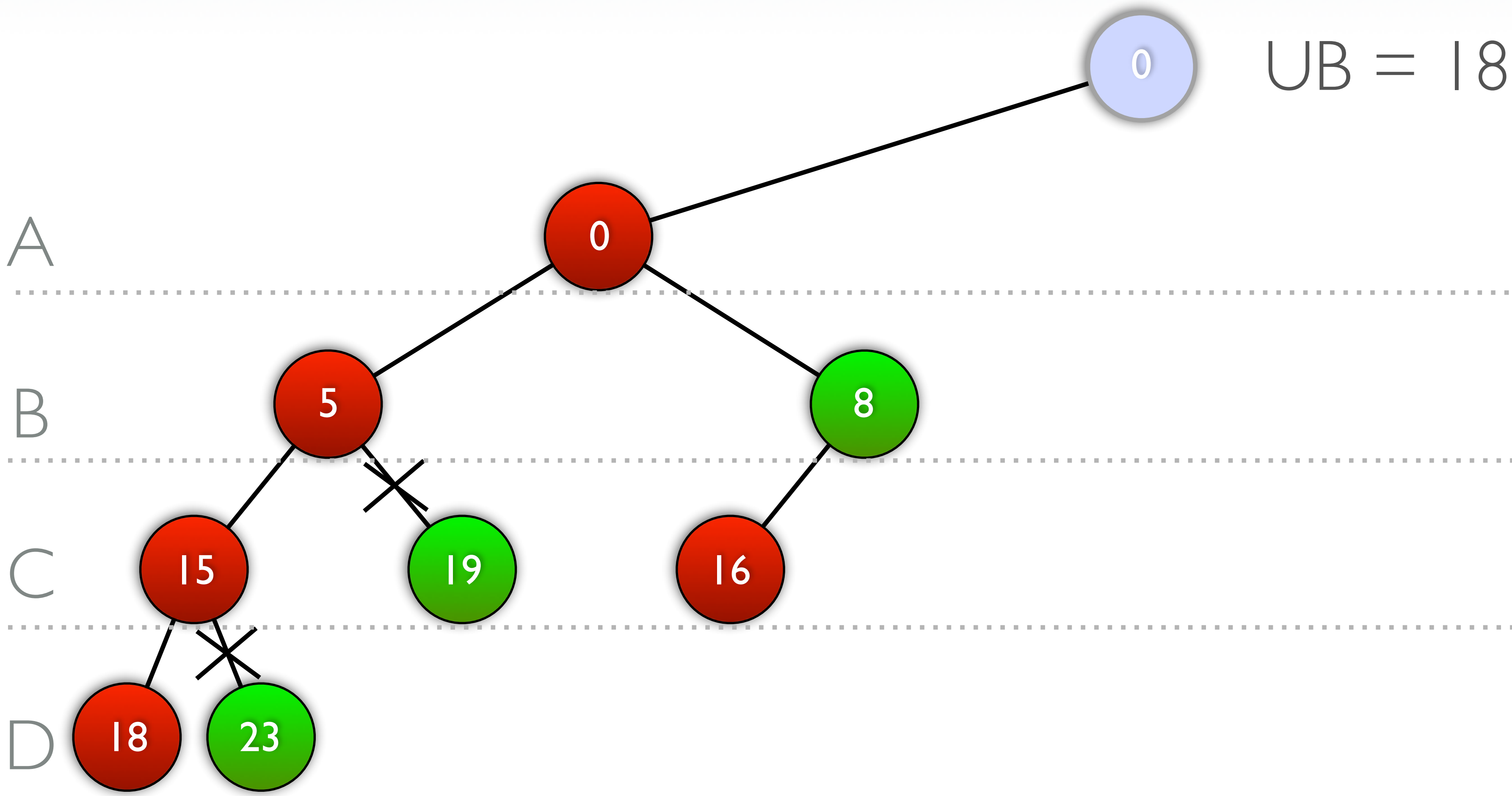
SBB



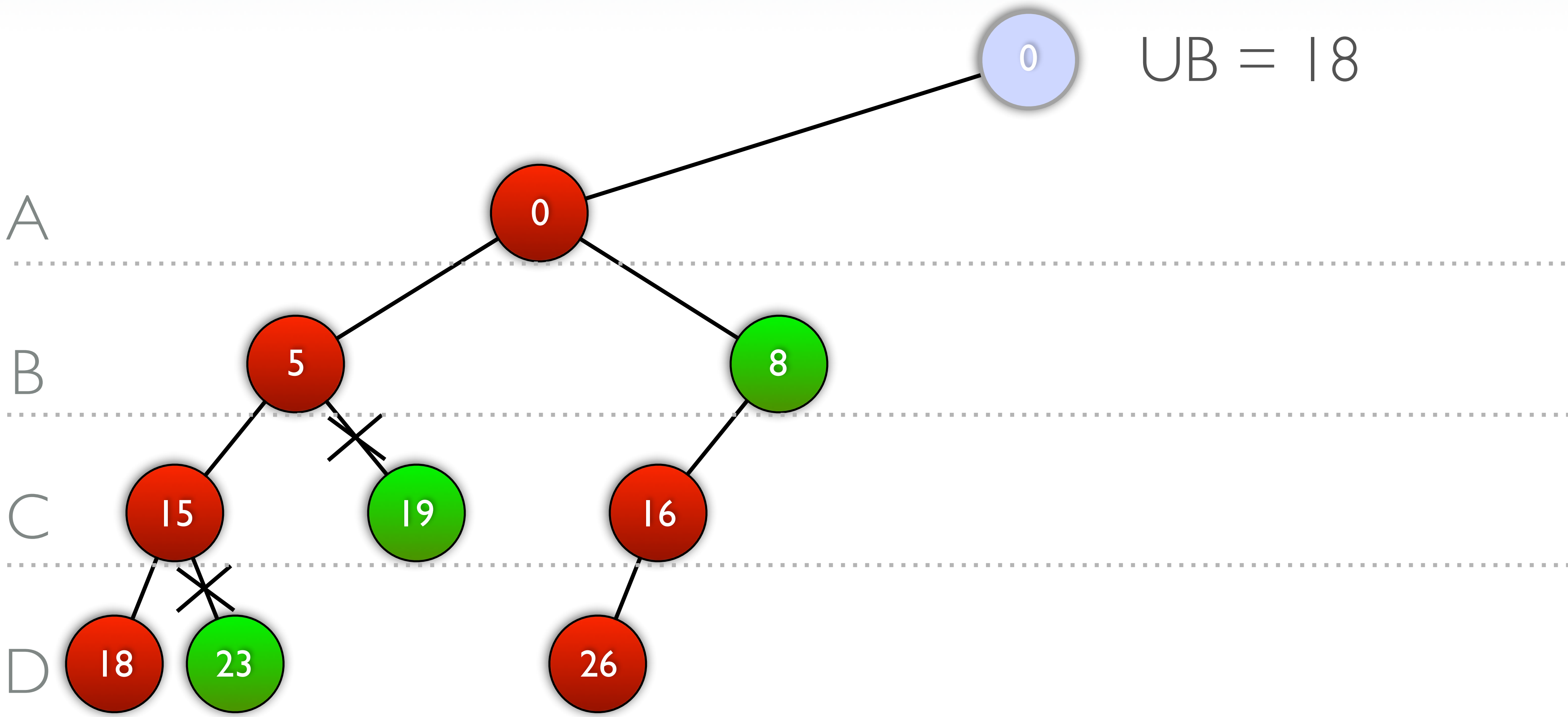
SBB



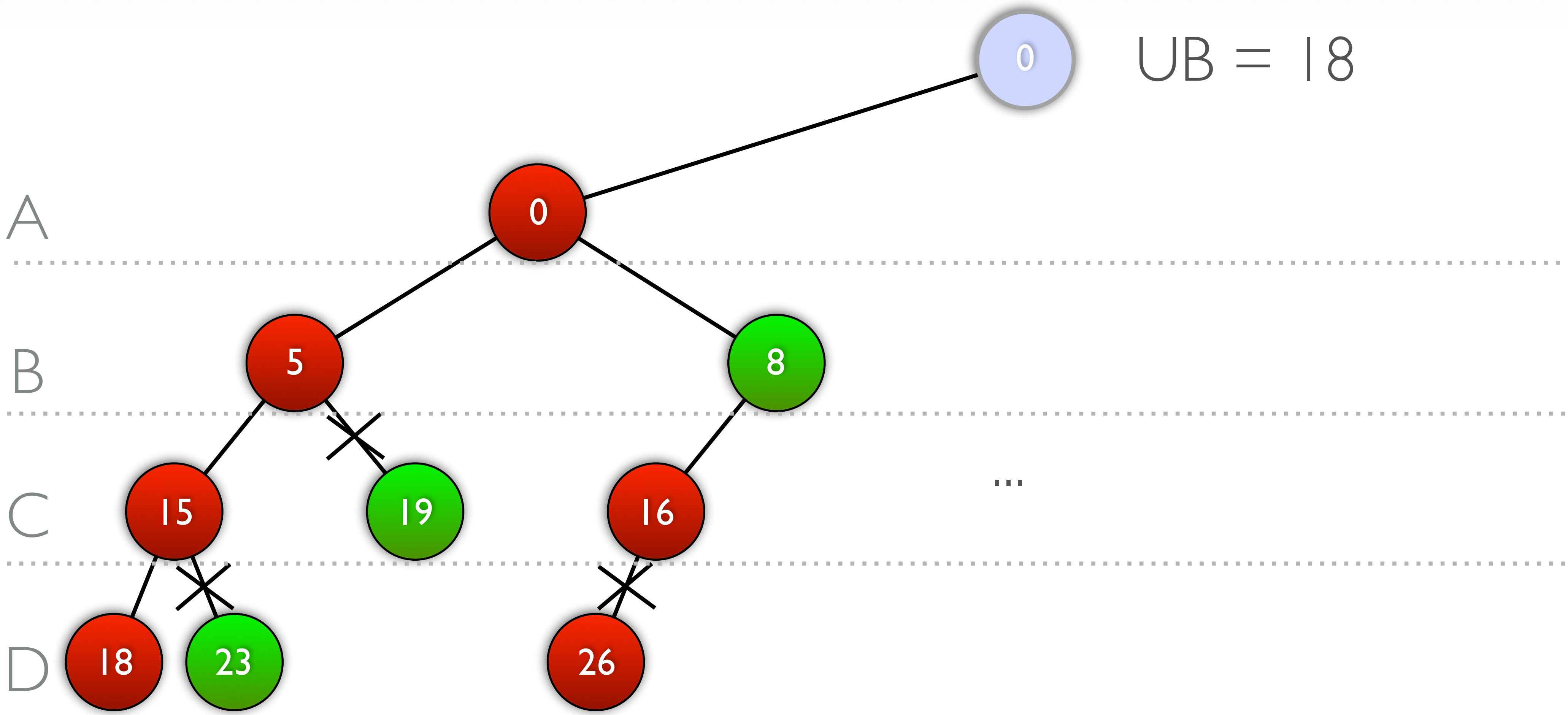
SBB



SBB



SBB

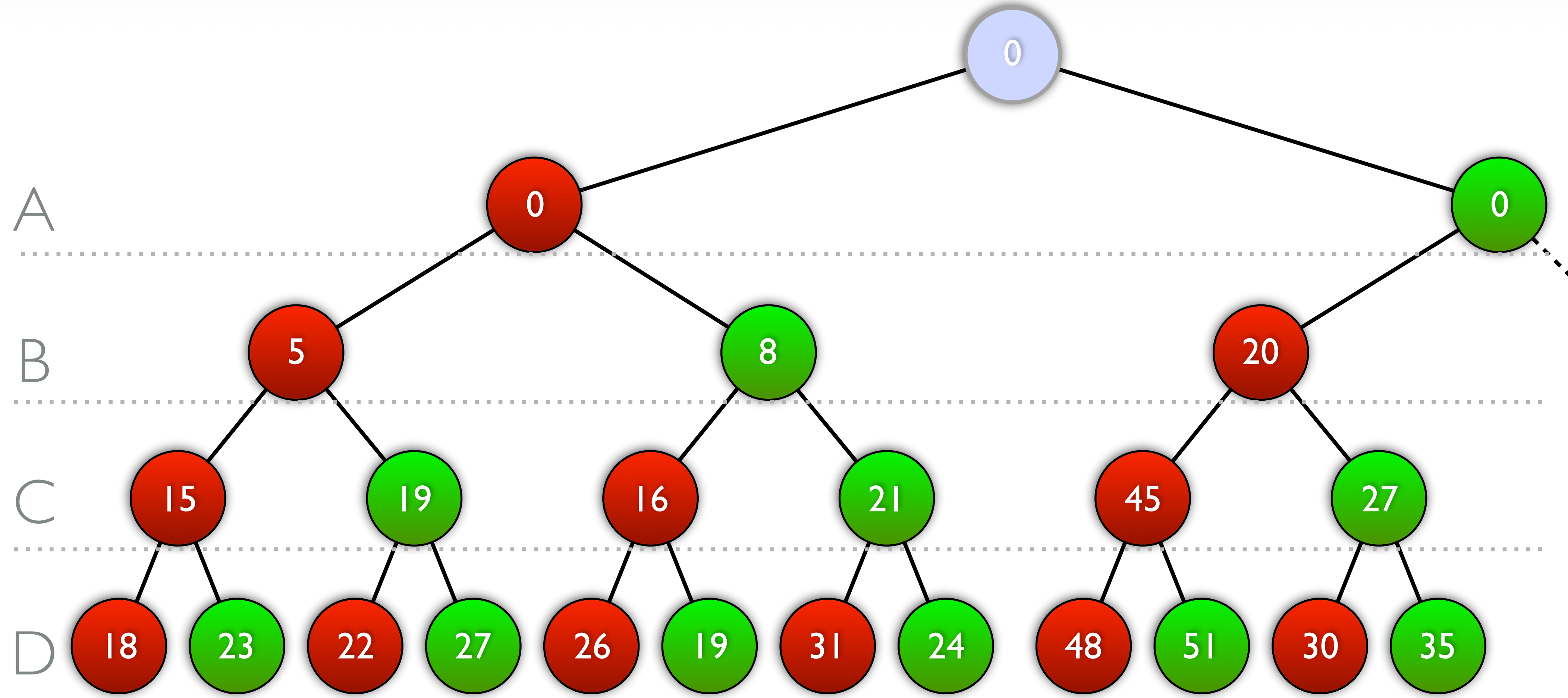


SBB

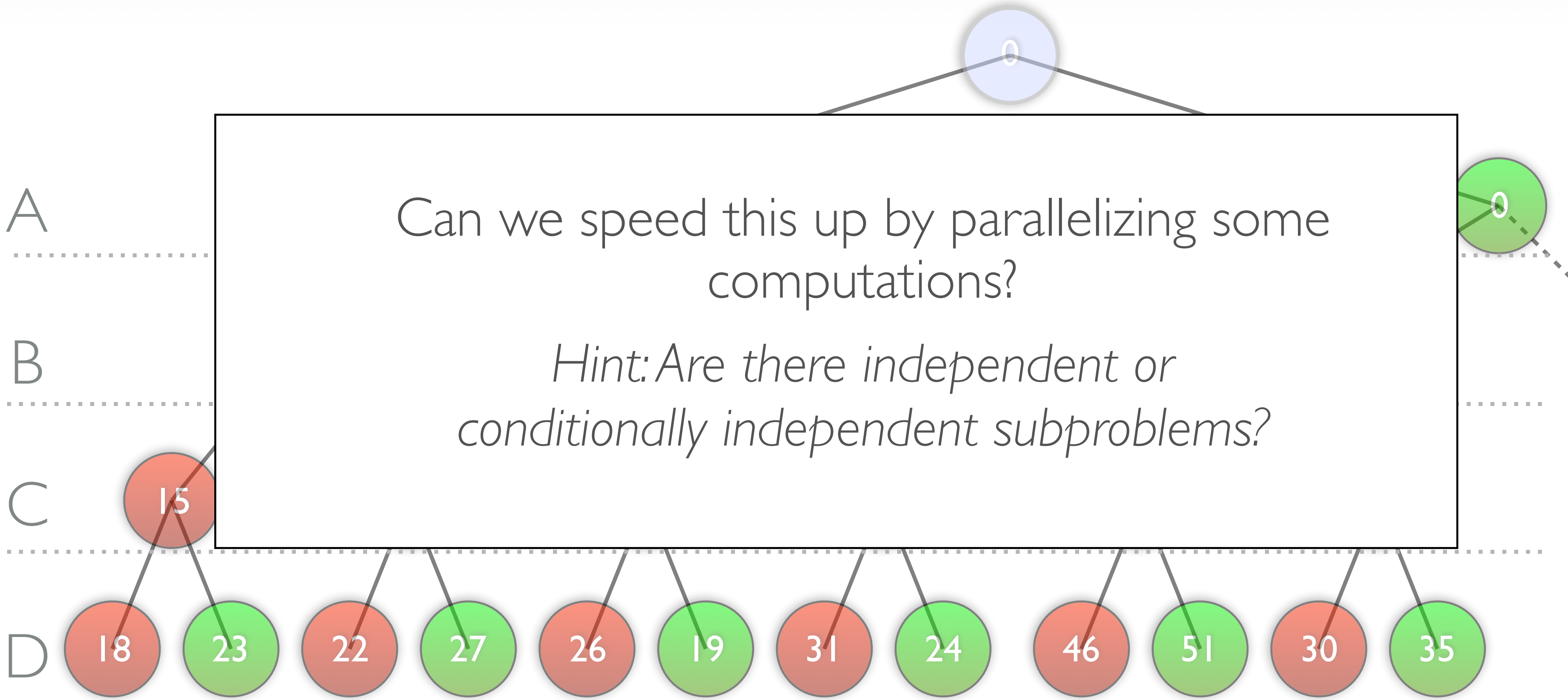
	SBB
Correct the solution it finds is optimal	Yes
Complete it terminates	Yes
Message Complexity max size of a message	$O(d)$
Network Load max number of messages	$O(b^d)$
Runtime	$O(b^d)$

branching factor = b
num variables = d

SBB



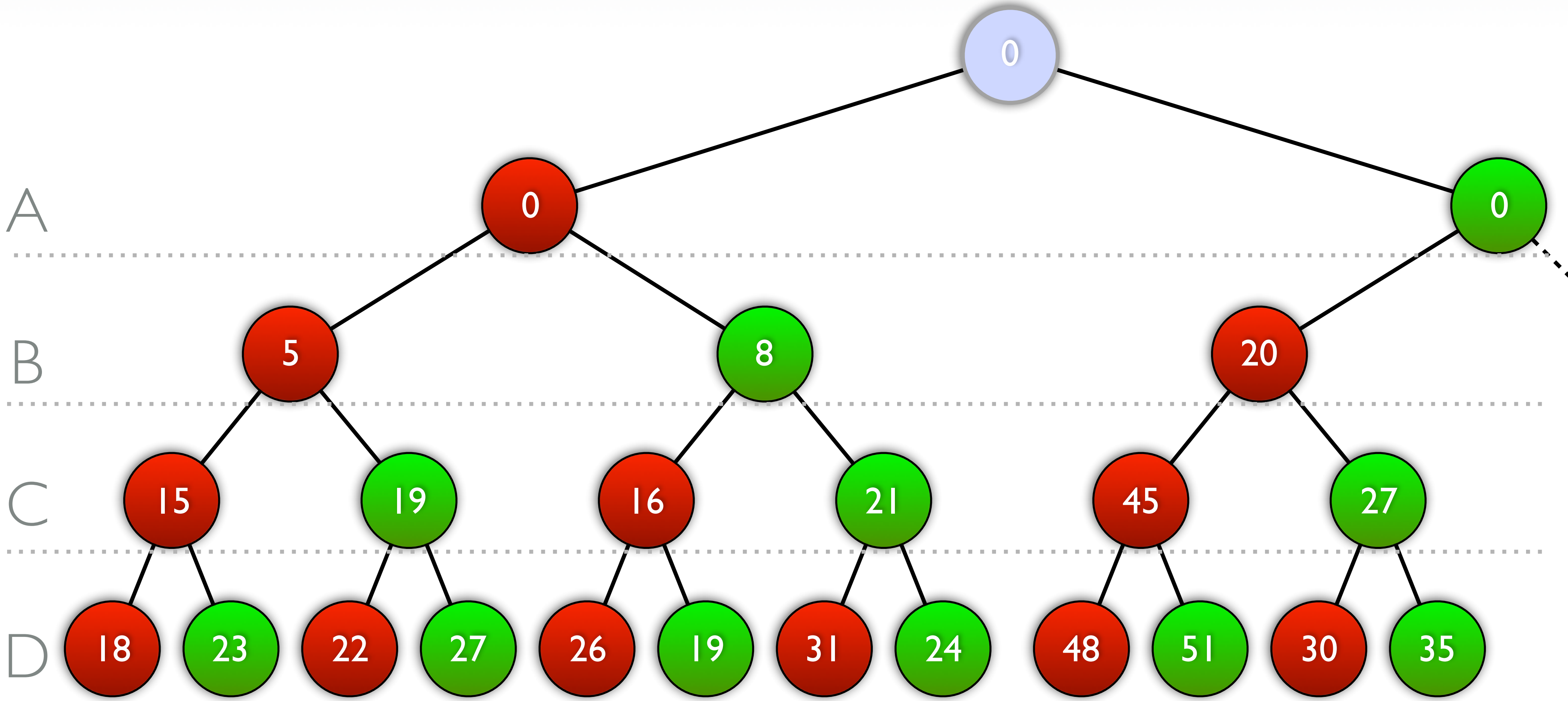
SBB



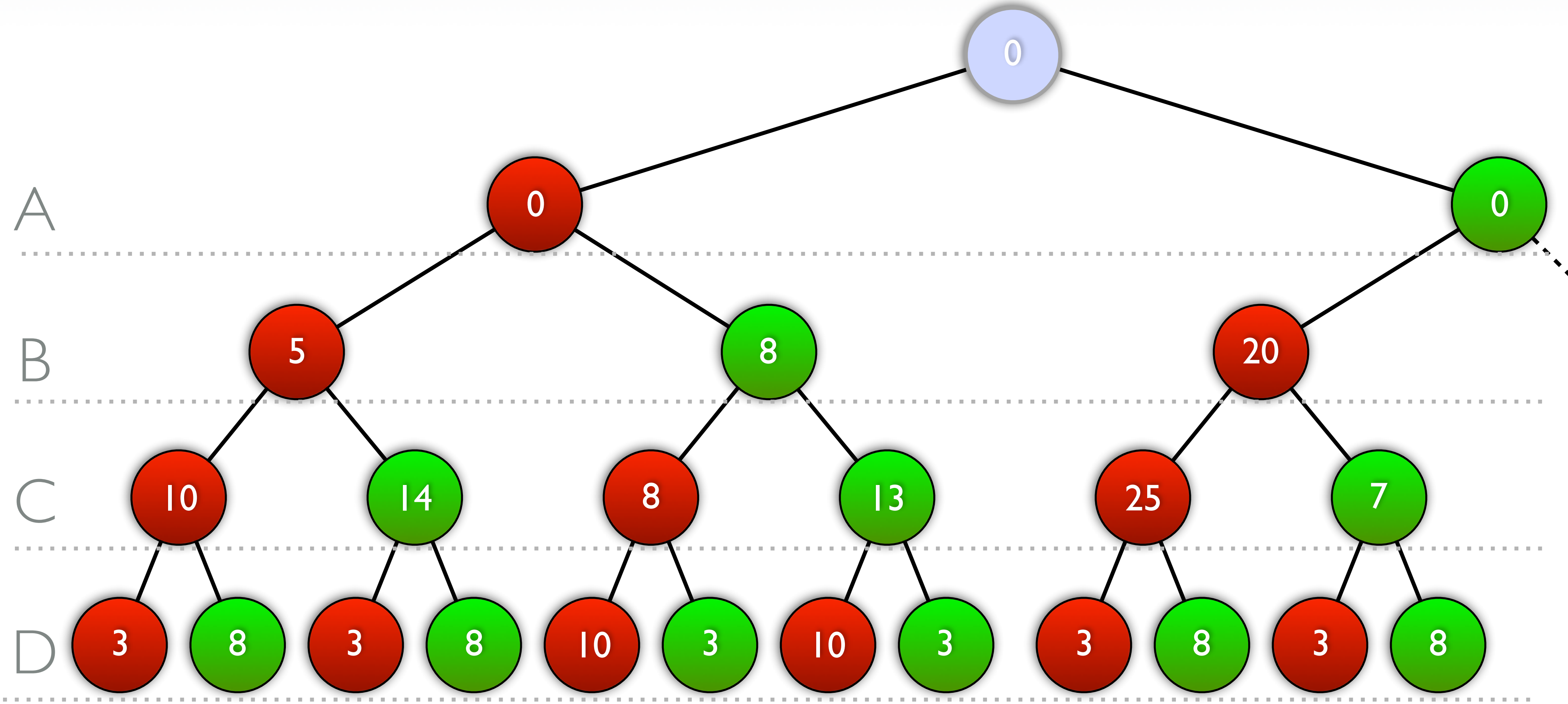
Can we speed this up by parallelizing some computations?

Hint: Are there independent or conditionally independent subproblems?

SBB

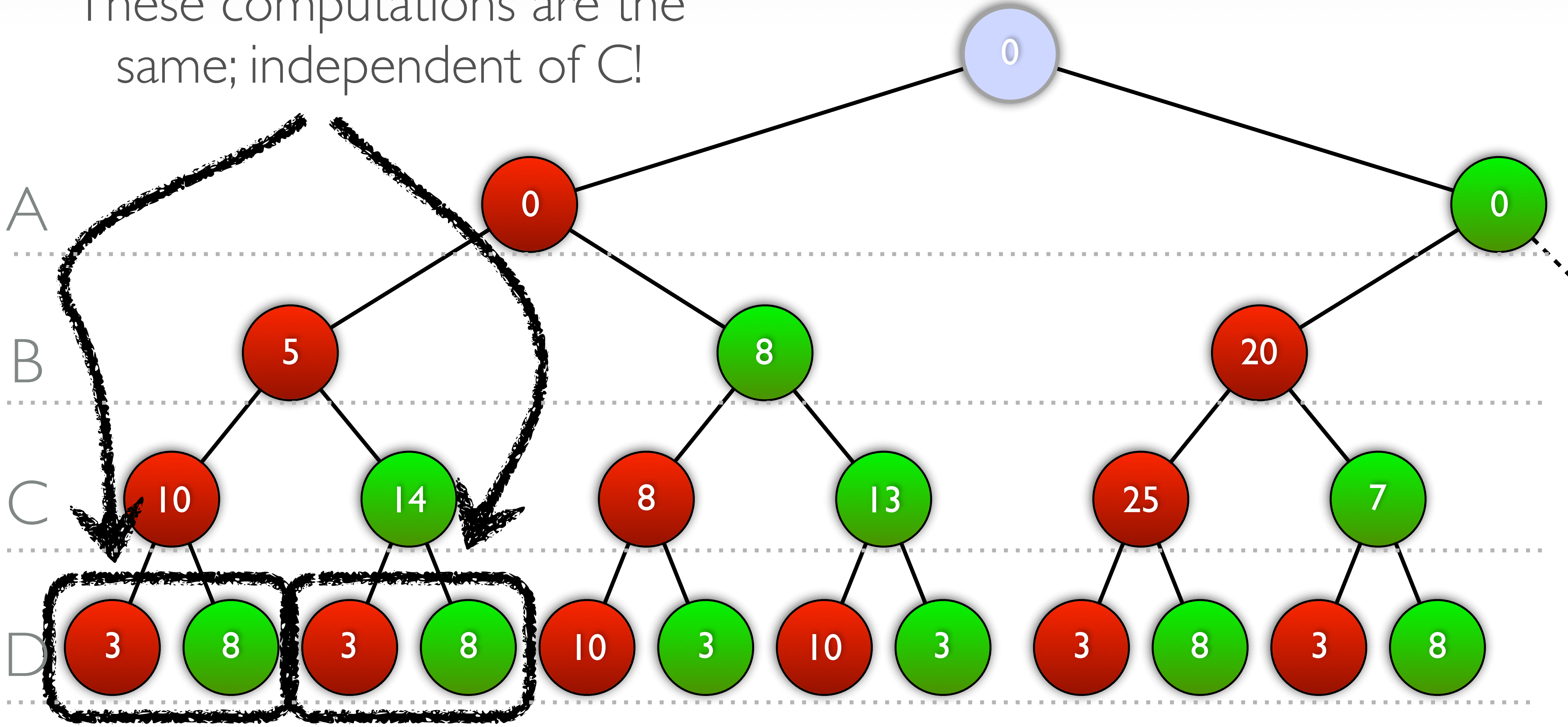


SBB

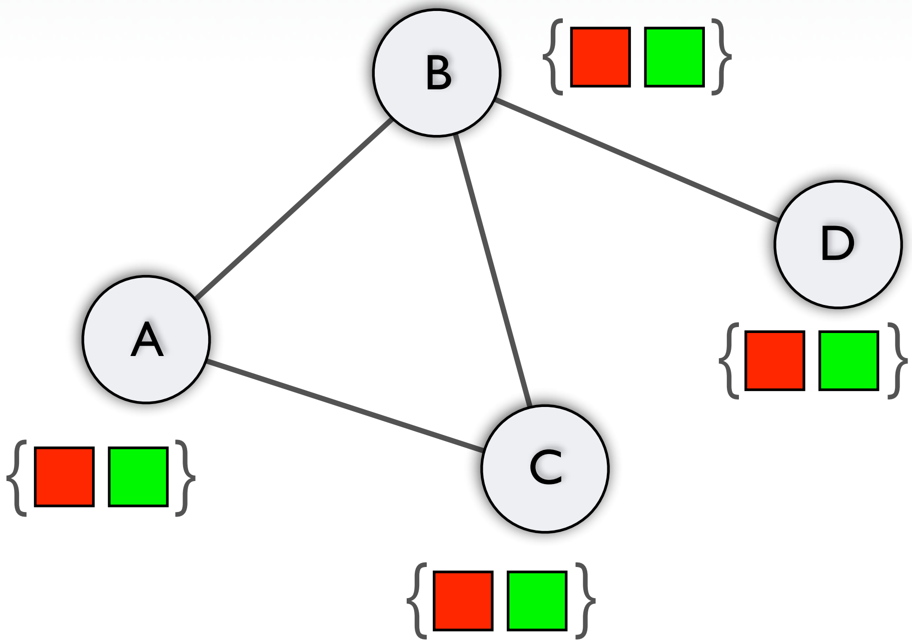


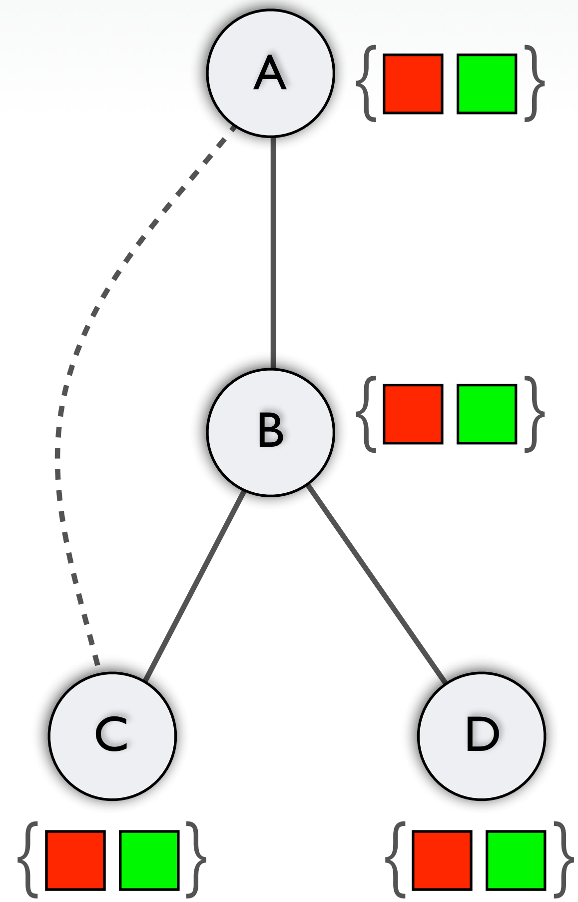
SBB

These computations are the same; independent of C!



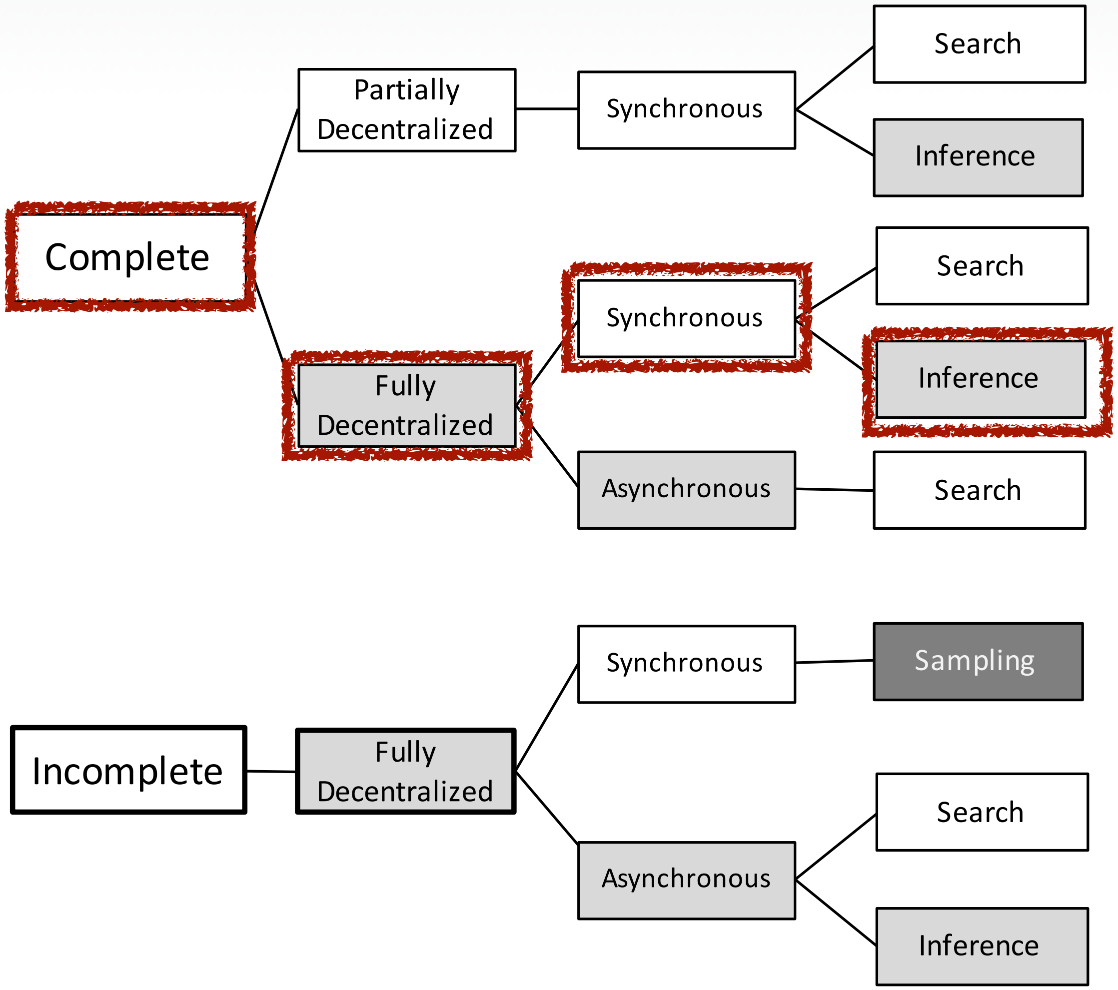
Pseudo-Tree





Definition: *A spanning tree of the constraint graph such that no two nodes in sibling subtrees share a constraint in the constraint graph*

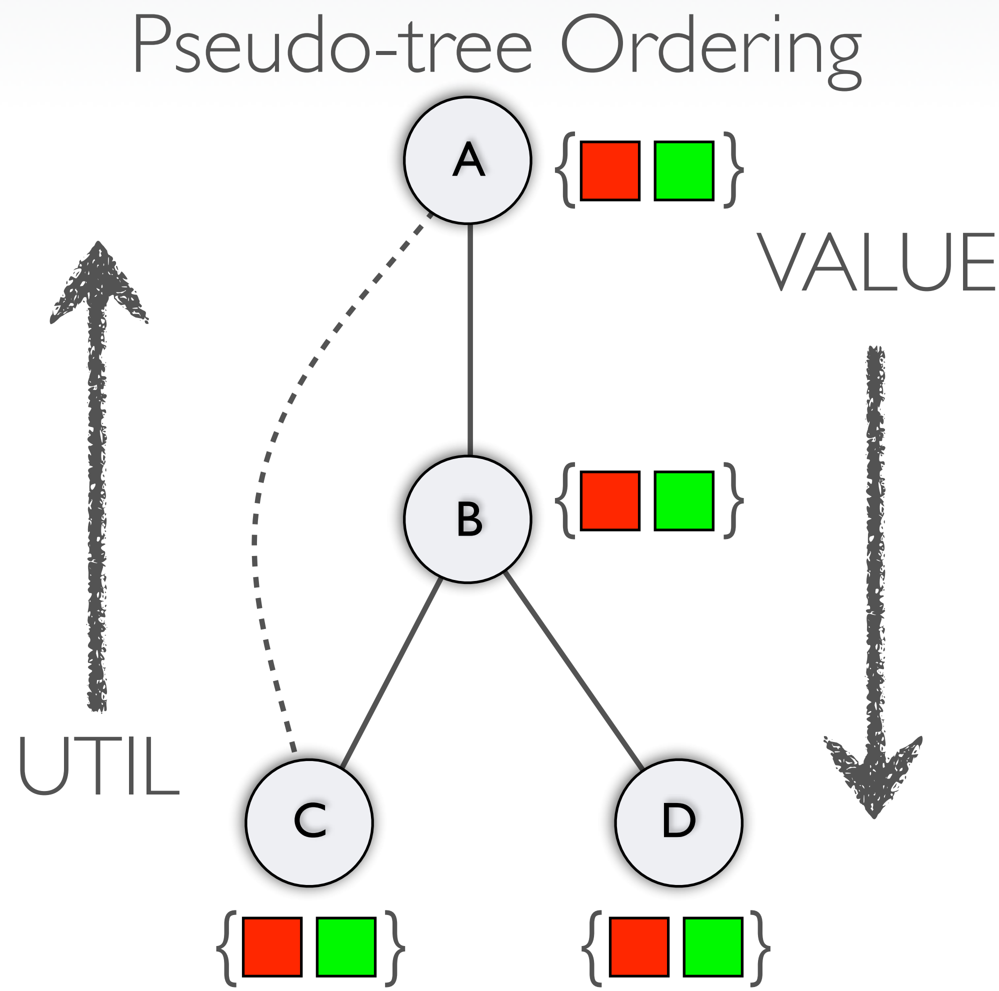
DCOP Algorithms



Distributed Pseudotree Optimization Procedure (DPOP)

DPOP

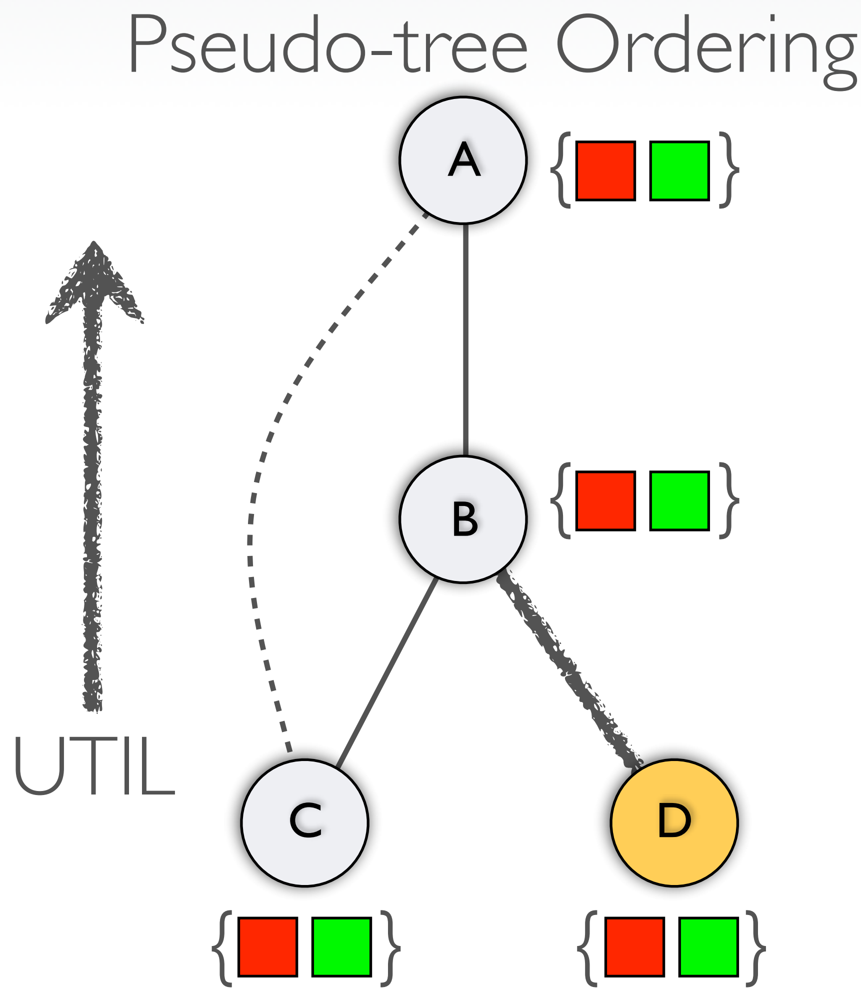
- Extension of the Bucket Elimination (BE)
- Agents operate on a pseudo-tree ordering
- UTIL phase: Leaves to root
- VALUE phase: Root to leaves



Adrian Petcu, Boi Faltings: A Scalable Method for Multiagent Constraint Optimization. IJCAI 2005: 266-271

DPOP

B	D	(B,D)
r	r	3
r	g	8
g	r	10
g	g	3



DPOP

B	D	(B,D)
r	r	3
r	g	8
g	r	10
g	g	3

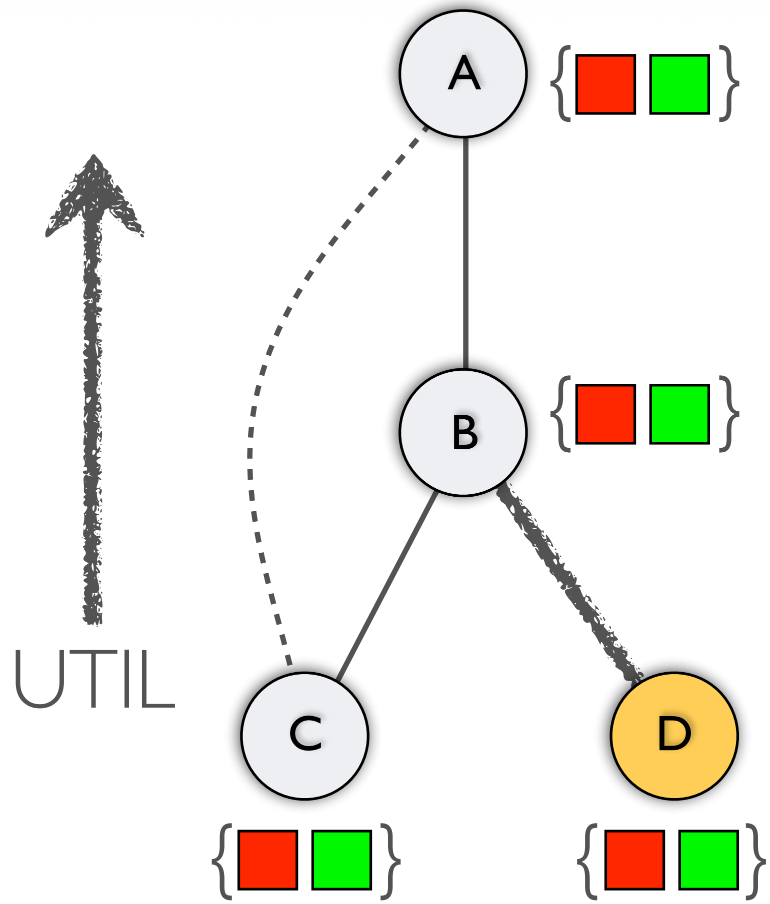
$\min\{3, 8\} = 3$

$\min\{10, 3\} = 3$

MSG to B

B	cost
r	3
g	3

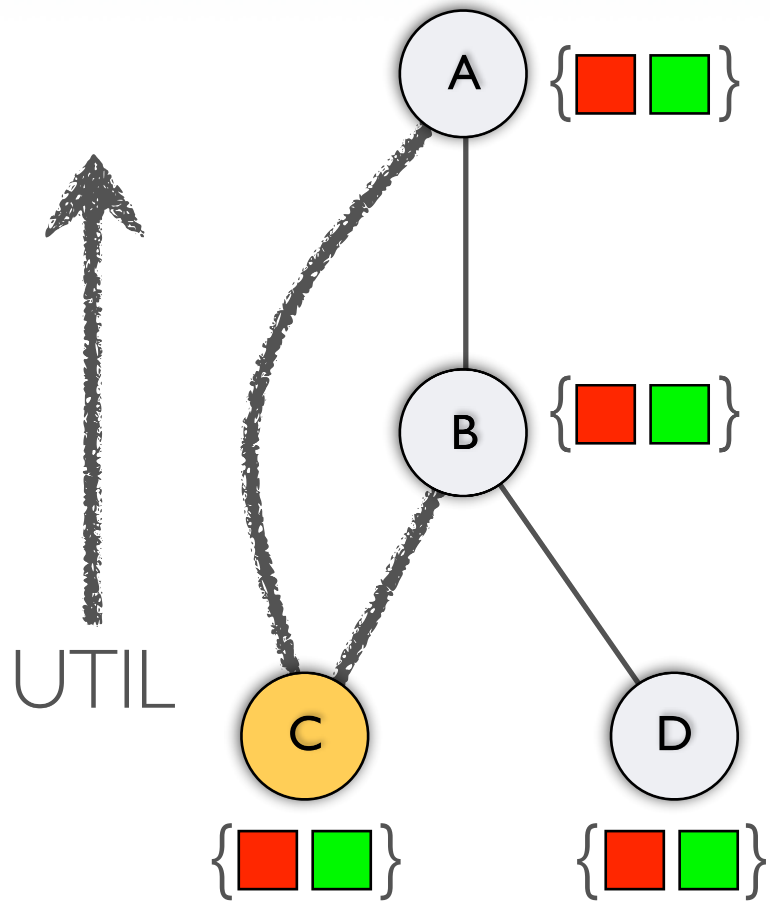
Pseudo-tree Ordering



DPOP

A	B	C	(B,C)	(A,C)	
r	r	r	5	5	10
r	r	g	4	8	12
r	g	r	3	5	8
r	g	g	3	8	11
g	r	r	5	10	15
g	r	g	4	3	7
g	g	r	3	10	13
g	g	g	3	3	6

Pseudo-tree Ordering

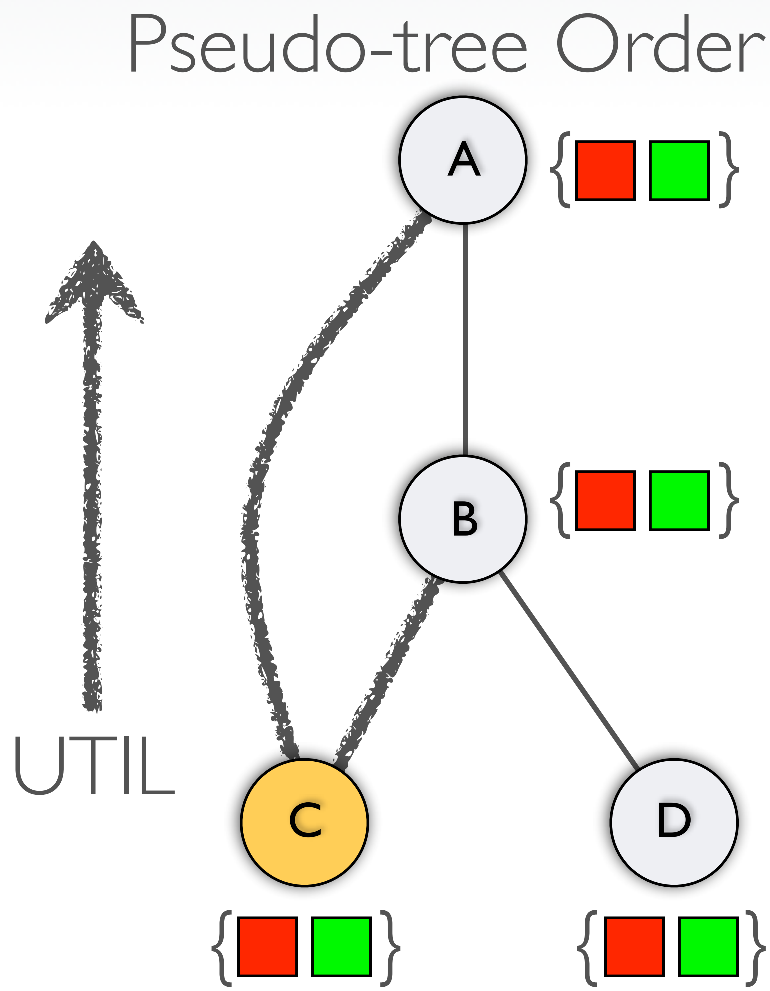


DPOP

A	B	C	(B,C)	(A,C)	
r	r	r	5	5	10
r	r	g	4	8	12
r	g	r	3	5	8
r	g	g	3	8	11
g	r	r	5	10	15
g	r	g	4	3	7
g	g	r	3	10	13
g	g	g	3	3	6

MSG to B

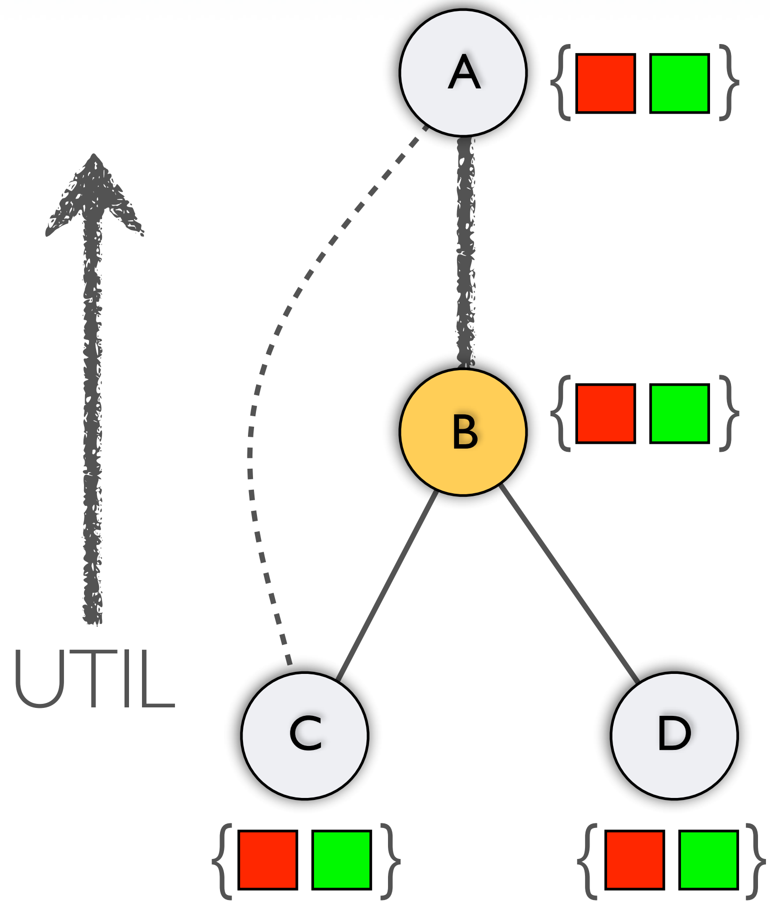
A	B	
r	r	10
r	g	8
g	r	7
g	g	6



DPOP

A	B	(A,B)	Util C	Util D	
r	r	5	10	3	18
r	g	8	8	3	19
g	r	20	7	3	30
g	g	3	6	3	12

Pseudo-tree Ordering



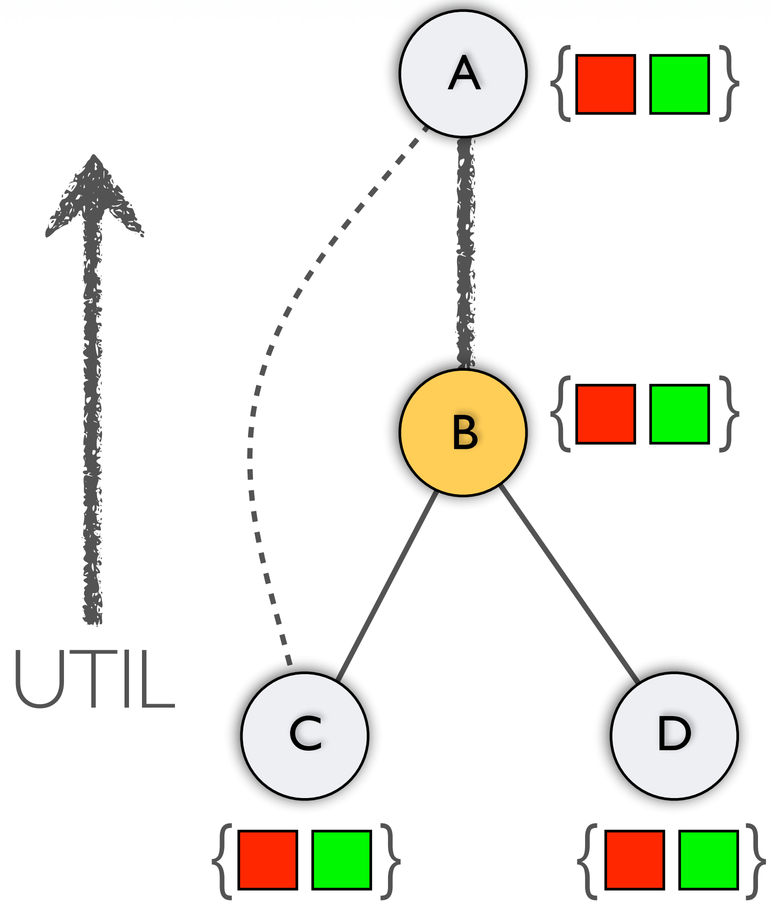
DPOP

A	B	(A,B)	Util C	Util D	
r	r	5	10	3	18
r	g	8	8	3	19
g	r	20	7	3	30
g	g	3	6	3	12

MSG to A

A	cost
r	18
g	12

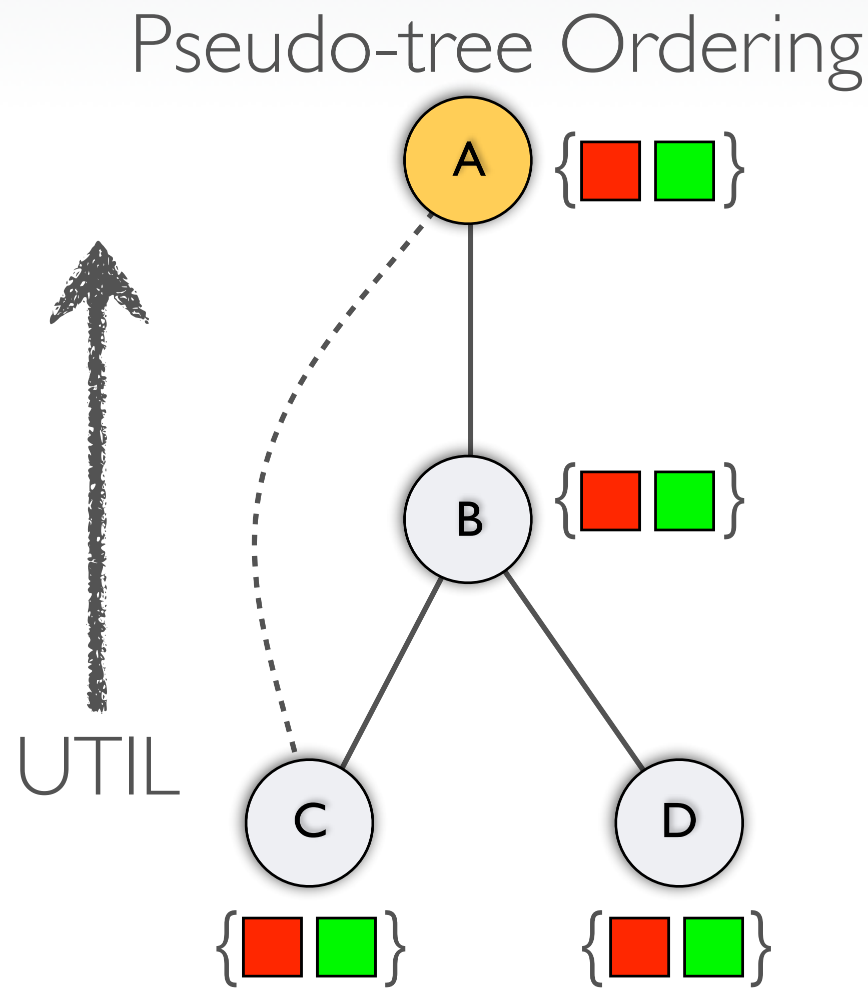
Pseudo-tree Ordering



DPOP

A	cost
r	18
g	12

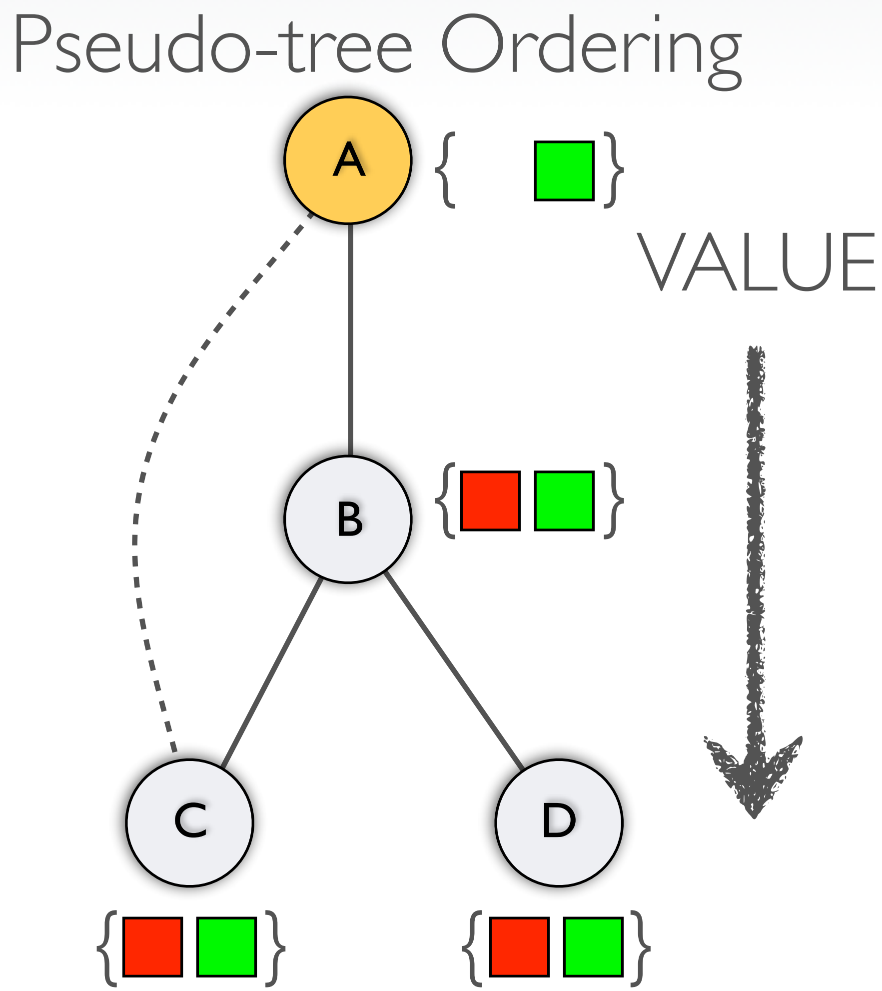
optimal cost = 12



DPOP

A	cost
r	18
g	12

- Select value for A = 'g'
- Send MSG A = 'g' to agents B and C

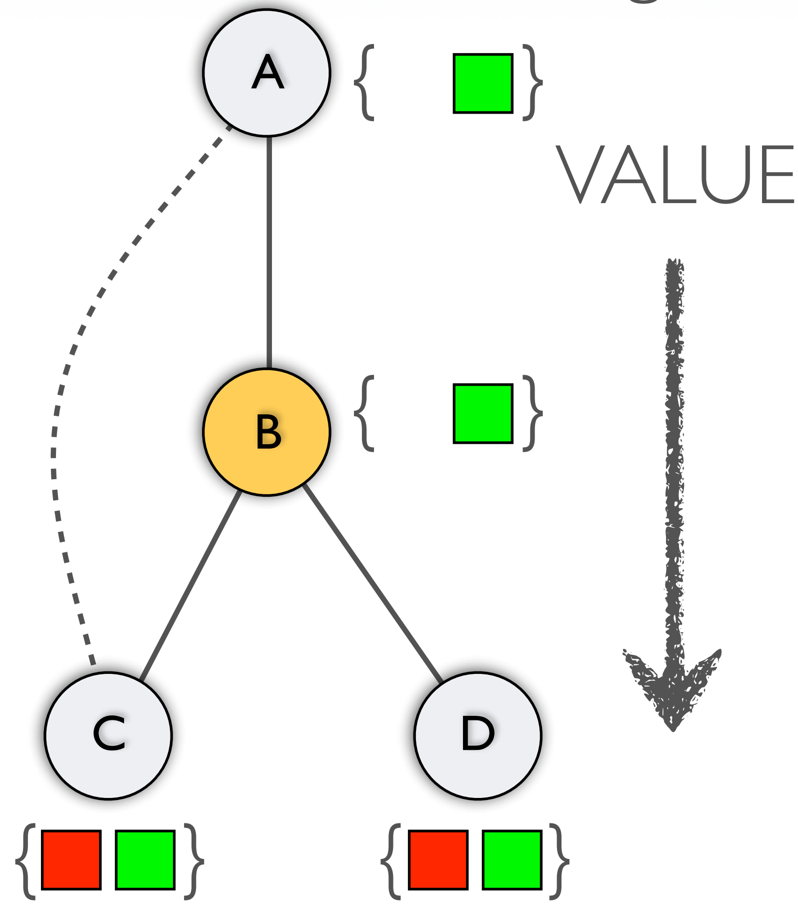


DPOP

A	B	(A,B)	Util C	Util D	
r	r	5	10	3	18
r	g	8	8	3	19
g	r	20	7	3	30
g	**g**	3	6	3	12

- Select value for B = 'g'
- Send MSG B = 'g' to agents C and D

Pseudo-tree Ordering

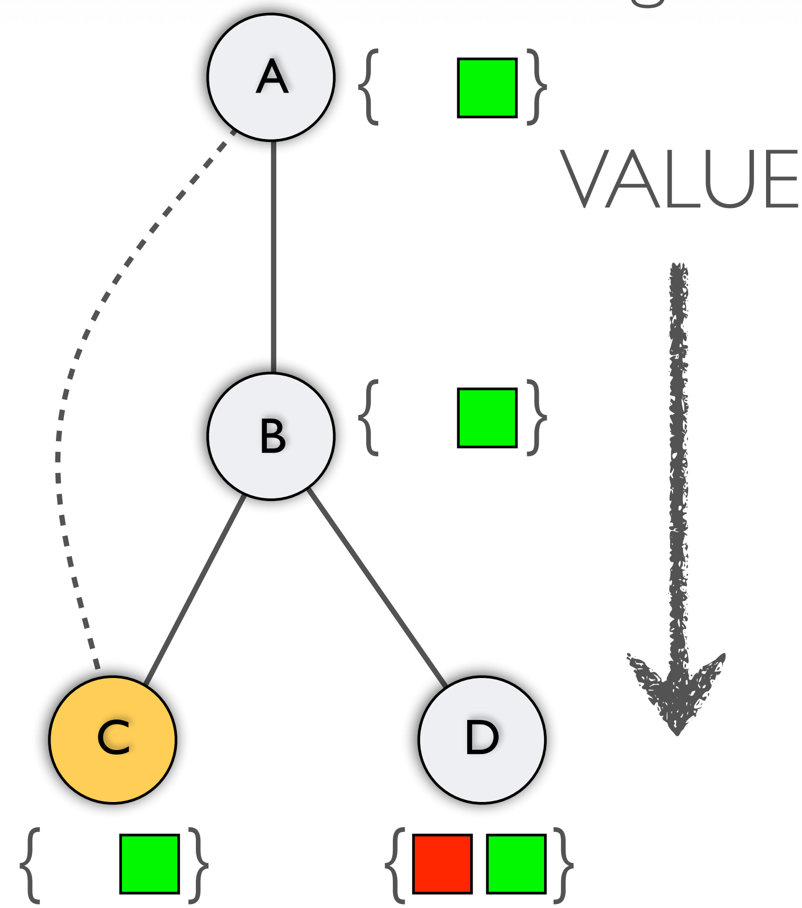


DPOP

A	B	C	(B,C)	(A,C)	
r	r	r	5	5	10
r	r	g	4	8	12
r	g	r	3	5	8
r	g	g	3	8	11
g	r	r	5	10	15
g	r	g	4	3	7
g	g	r	3	10	13
g	g	**g**	3	3	6

- Select value for C = ‘g’

Pseudo-tree Ordering

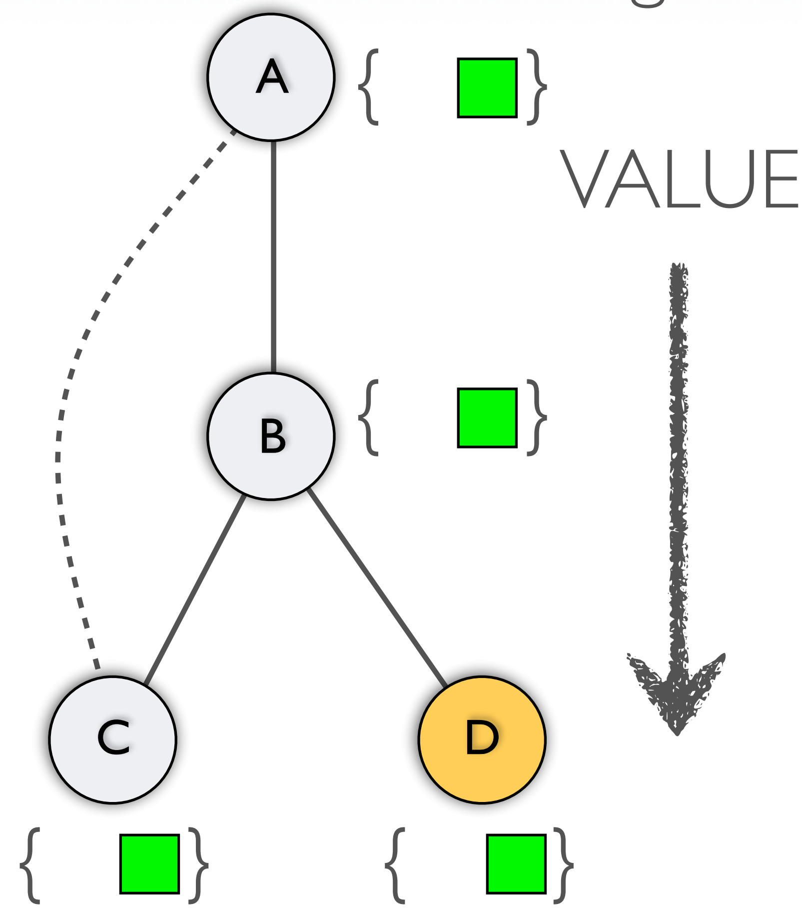


DPOP

B	D	(B,D)
r	r	3
r	g	8
g	r	10
g	**g**	3

- Select value for D = ‘g’

Pseudo-tree Ordering



DPOP

	SBB	DPOP
Correct the solution it finds is optimal	Yes	Yes
Complete it terminates	Yes	Yes
Message Complexity max size of a message	$O(d)$	$O(b^d)$
Network Load max number of messages	$O(b^d)$	$O(d)$
Runtime	$O(b^d)$	$O(b^d)$

branching factor = b
num variables = d

Critical Overview

Search Algorithms

Inference Algorithms

increasing memory

polynomial

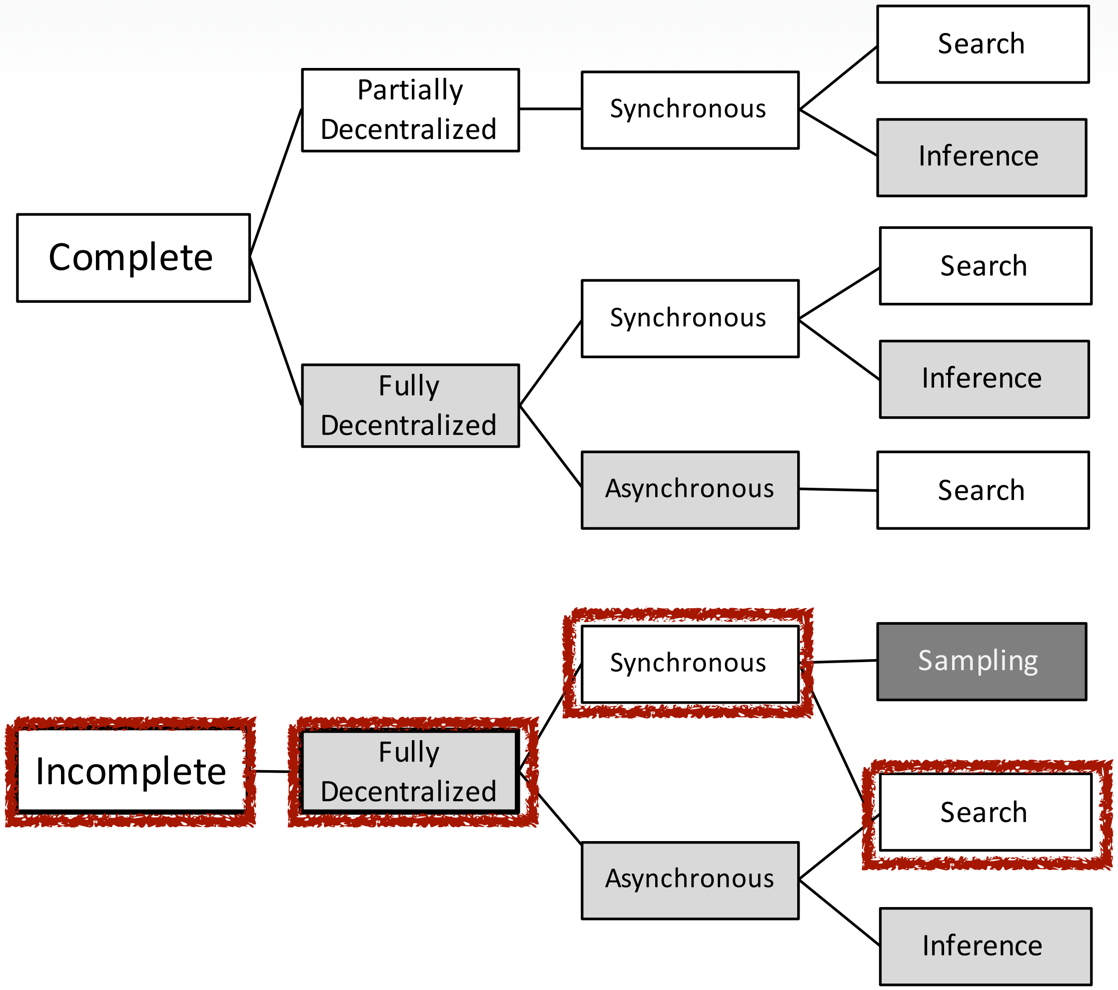
exponential

decreasing network load

exponential

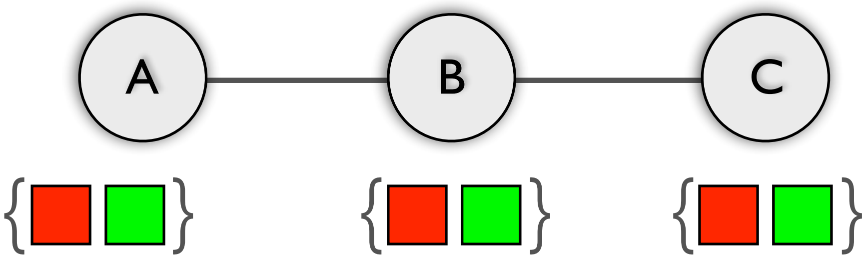
polynomial

DCOP Algorithms



Distributed Local Search

Local Search Algorithms



x_i	x_j	Utility (A,B)	Utility (B,C)
		5	5
		0	0
		0	0
		8	8

Weixiong Zhang, Guandong Wang, Zhao Xing, Lars Wittenburg: Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks. Artif. Intell. 161(1-2): 55-87 (2005)
Rajiv Maheswaran, Jonathan Pearce, Milind Tambe: Distributed Algorithms for DCOP: A Graphical-Game-Based Approach. ISCA PDCS 2004: 432-439

Local Search Algorithms

- DSA: Distributed Stochastic Algorithm
- MGM: Maximum Gain Messages Algorithm

- Every agent individually decides whether to change its value or not
- Decision involves
  - knowing neighbors' values
  - calculation of utility gain by changing values
  - probabilities

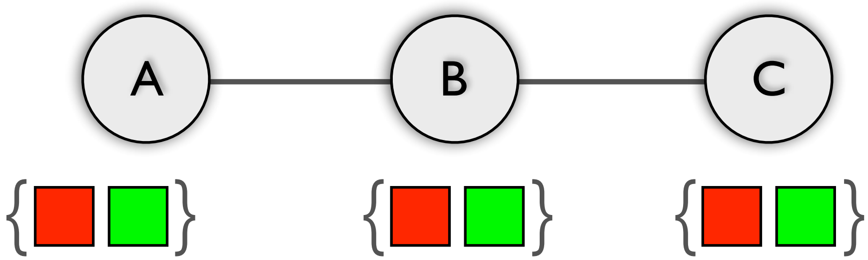
Weixiong Zhang, Guandong Wang, Zhao Xing, Lars Wittenburg: Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks. Artif. Intell. 161(1-2): 55-87 (2005)
Rajiv Maheswaran, Jonathan Pearce, Milind Tambe: Distributed Algorithms for DCOP: A Graphical-Game-Based Approach. ISCA PDCS 2004: 432-439

DSA Algorithm

- All agents execute the following
  - Randomly choose a value
  - while (termination is not met)
    - if (a new value is assigned)
      - send the new value to neighbors
    - collect neighbors' new values if any
    - select and assign the next value based on assignment rule

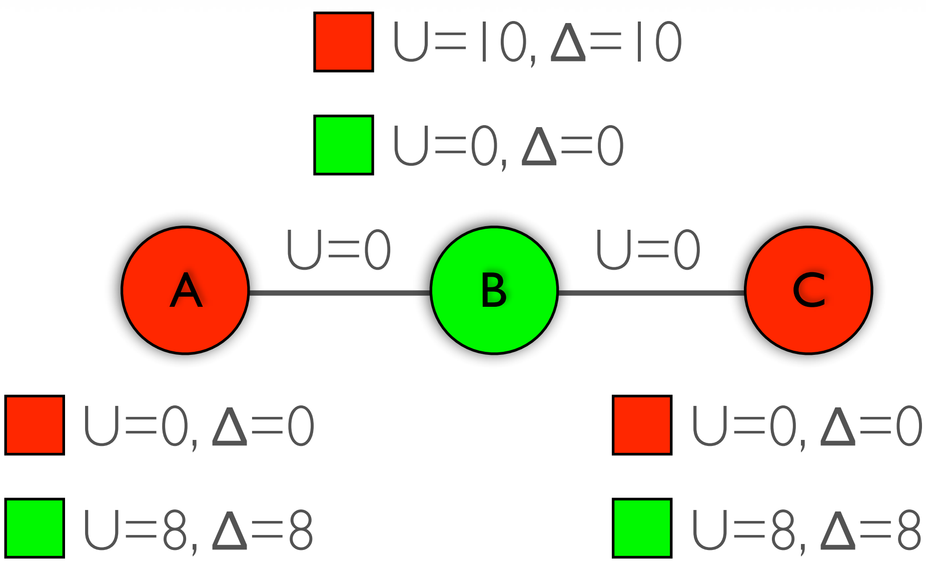
Weixiong Zhang, Guandong Wang, Zhao Xing, Lars Wittenburg: Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks. Artif. Intell. 161(1-2): 55-87 (2005)

DSA Algorithm



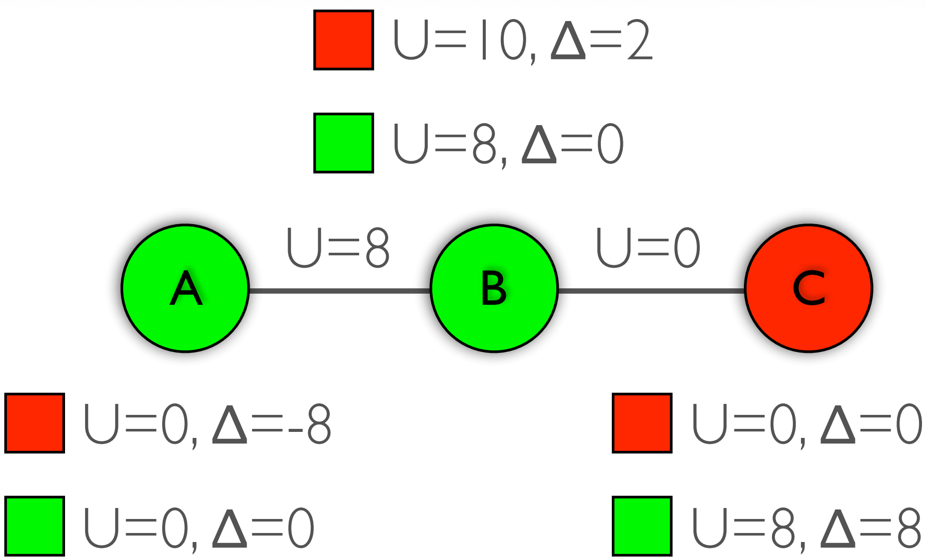
x_i	x_j	*Utility (A,B)*	*Utility (B,C)*
		5	5
		0	0
		0	0
		8	8

DSA Algorithm



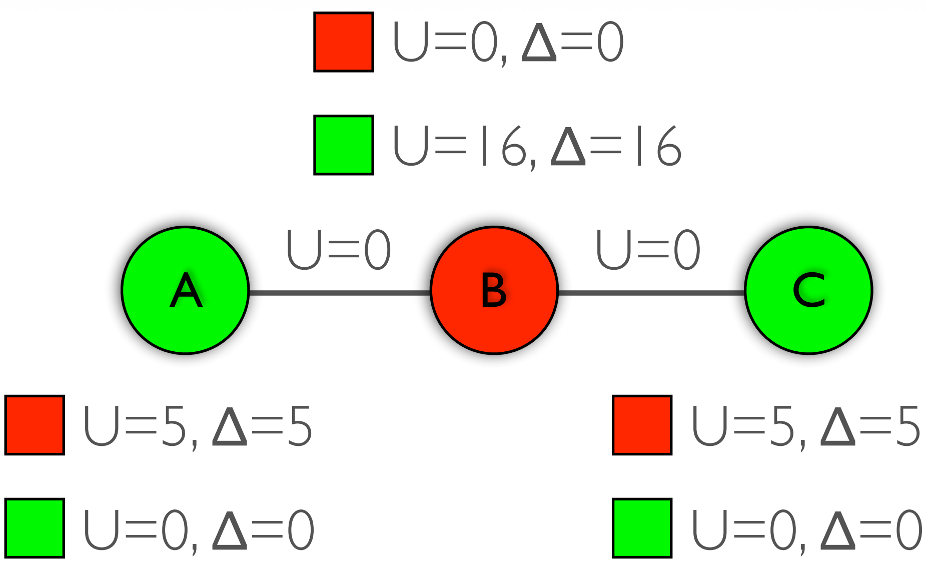
x_i	x_j	*Utility (A,B)*	*Utility (B,C)*
		5	5
		0	0
		0	0
		8	8

DSA Algorithm



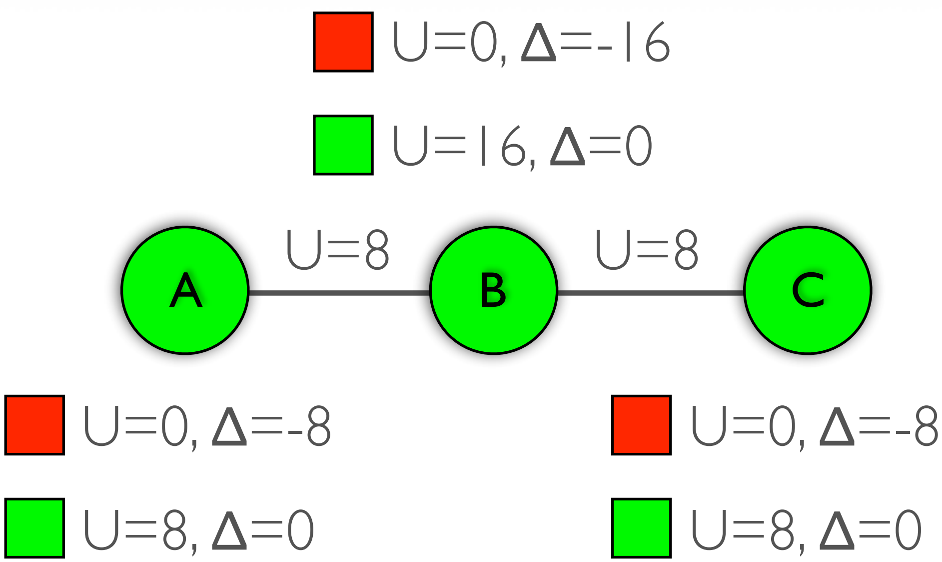
x_i	x_j	*Utility (A,B)*	*Utility (B,C)*
		5	5
		0	0
		0	0
		8	8

DSA Algorithm



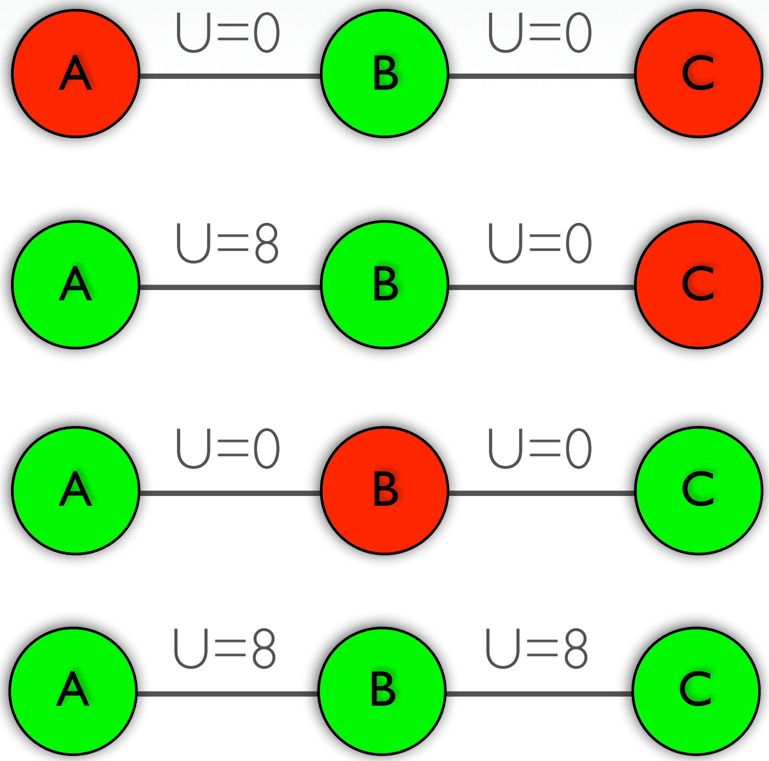
x_i	x_j	*Utility (A,B)*	*Utility (B,C)*
		5	5
		0	0
		0	0
		8	8

DSA Algorithm



x_i	x_j	*Utility (A,B)*	*Utility (B,C)*
		5	5
		0	0
		0	0
		8	8

DSA Algorithm



One possible execution trace

x_i	x_j	*Utility (A,B)*	*Utility (B,C)*
		5	5
		0	0
		0	0
		8	8

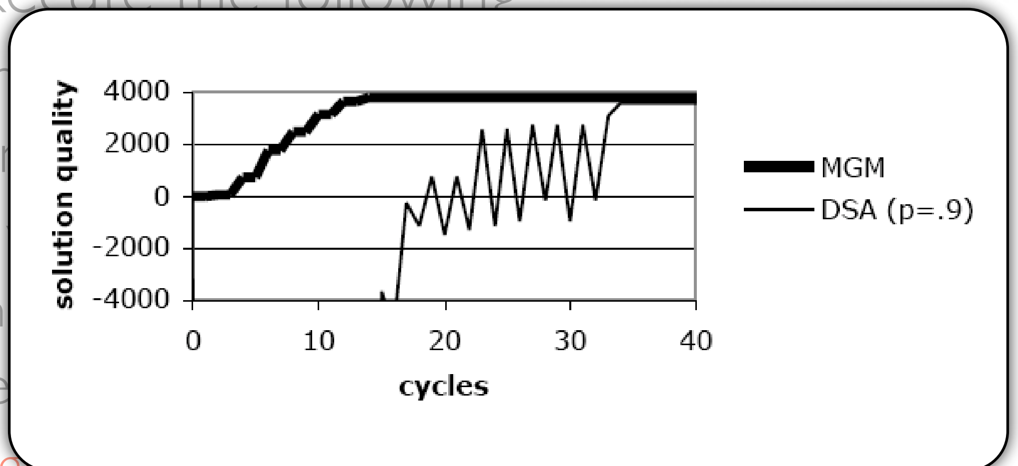
MGM Algorithm

- All agents execute the following
  - Randomly choose a value
  - while (termination is not met)
    - if (a new value is assigned)
      - send the new value to neighbors
    - collect neighbors' new values if any
    - *calculate gain and send it to neighbors*
    - *collect neighbors' gains*
    - *if (it has the highest gain among all neighbors)*
      - *change value to the value that maximizes gain*

Rajiv Maheswaran, Jonathan Pearce, Milind Tambe: Distributed Algorithms for DCOP: A Graphical-Game-Based Approach. ISCA PDCS 2004: 432-439

MGM Algorithm

- All agents execute the following
  - Randomly ch
  - while (termi
    - if (a new
      - send th
    - collect ne
    - *calculate gain and send it to neighbors*



Great if you need an anytime algorithm

      - *change value to the value that maximizes gain*

Rajiv Maheswaran, Jonathan Pearce, Milind Tambe: Distributed Algorithms for DCOP: A Graphical-Game-Based Approach. ISCA PDCS 2004: 432-439

DCOP Extensions

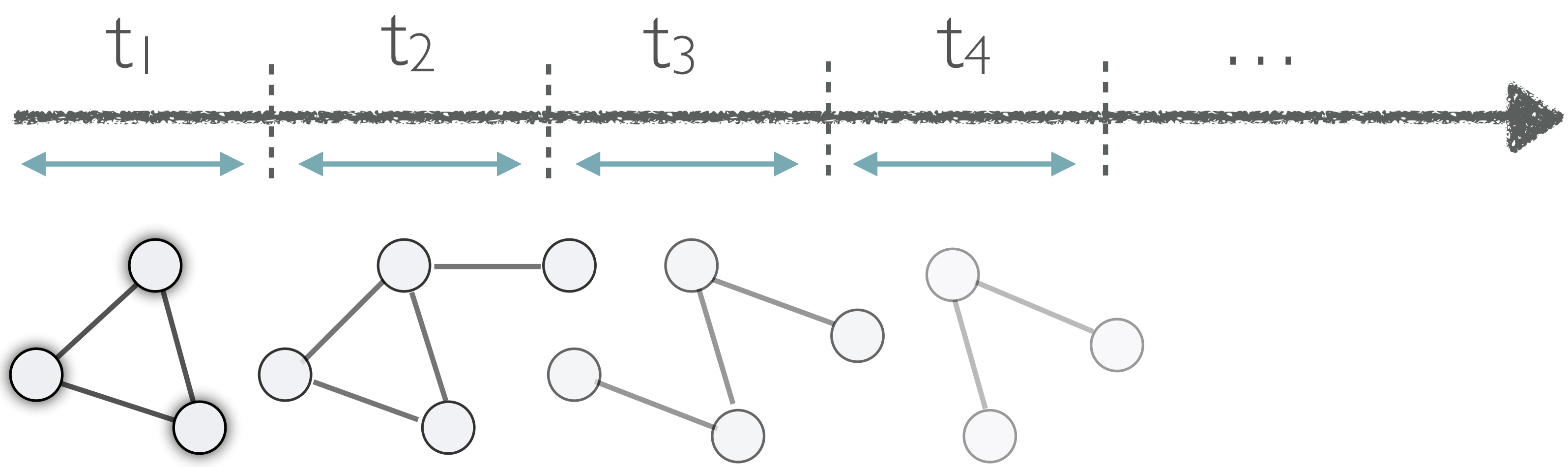
AAAI-20 Tutorial on
Multi-Agent Distributed Constrained Optimization

Dynamic DCOP

- Why dynamic DCOPs?
  - MAS commonly exhibit dynamic environments
  - The capture scenarios with:
    - Moving agents, change of constraints, change of preferences
    - Additional information become available during problem solving
  - Application domains: Sensor networks, cloud computing, smart home automation, …

Dynamic DCOP

- A Dynamic DCOP is sequence $P_1, P_2, \ldots, P_k$ of k DCOPs
- The agent knowledge about the environment is confined within each time step
- Each DCOP is solved sequentially



Dynamic DCOP

- Current proposals on Dynamic DCOPs have focused on balancing reactiveness vs. proactiveness
  - Reactive algorithms: React to changes of the environment as soon as they are observed
  - Proactive algorithms: Use predictions about future events to better react to the changes in the environment

Continuous DCOPs

- Solves problems in which variables domains are continuous
- Arbitrary constraints (e.g., non-convex)

Continuous DCOP algorithms

- Continuous Maxsum (CMS)

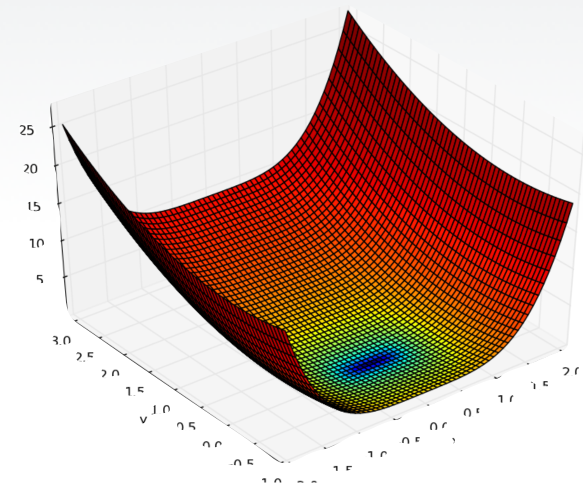
  Solve problems where constraints are linear piecewise functions
- Hybrid Continuous Maxsum (HCMS)

  No restriction on form of the functions

  Local optimization approach to improve quality
- C-DPOP (similar to DPOP but uses piecewise functions)

Distributed Convex Optimization

- Decentralized data and constraints
- Continuous variables
- Can describe many problems in optimization and learning



The convex function $f(x, y) = x^4 + y^2$.

- Consider a (centralized) problem of the form

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{s.t.} \quad & g_i(x) \leq 0 \quad i = 1, \ldots, m \\ & h_i(x) = 0 \quad i = 1, \ldots, p \end{aligned}$$

Dual Problem

- The *Lagrangian function* $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ is:

$$L(x, \nu, \lambda) = f(x) + \sum_{i=1}^{m} \nu_i g_i(x) + \sum_{i=1}^{p} \lambda_i h_i(x)$$

- Weighted sum of the constraint functions

Lagrangian Multipliers

- The *Lagrangian dual function* $LD : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}_{\geq 0}$ *is*

$$LD(\nu, \lambda) = \inf_{x \in \mathbb{R}^n} L(x, \nu, \lambda)$$

- Main advantage: it is concave

Dual Problem

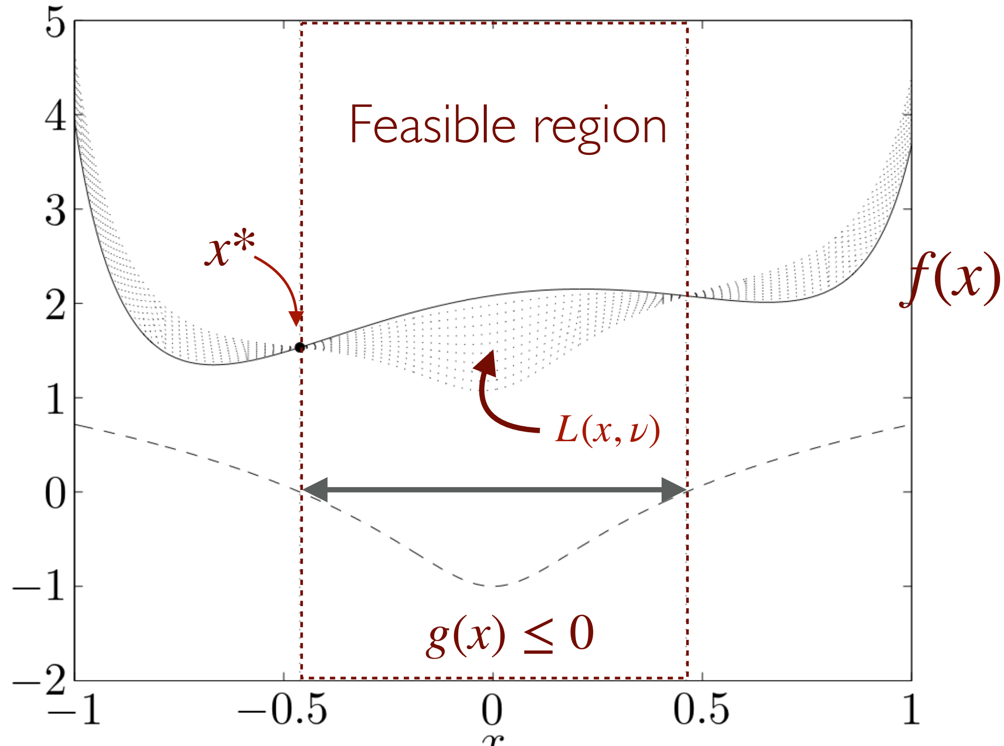
- Note that, for any feasible value $\tilde{x}$ we have

$$\sum_{i=1}^{m} \nu g_i(\tilde{x}) + \sum_{i=1}^{p} \lambda_i h_i(\tilde{x}) \leq 0$$

- That is

$$LD(\nu^*, \lambda^*) \leq L(\tilde{x}, \nu, \lambda) \leq f(\tilde{x})$$

for any feasible $\tilde{x}$



Each minimum value of $L(x, \nu)$ is less then or equal to $f(x)$

Dual Problem

- How to find the best lower bound?
- *Lagrangian dual problem*:
$$\max_{\nu,\lambda} \; LD(\nu, \lambda)$$
$$\text{s.t.} \;\; \nu \geq 0$$
- Optimal Lagrangian multipliers: $(\nu*, \lambda*)$
- Assuming strong duality: $x^* = \arg\min_x L(x, \nu^*, \lambda^*)$

Dual Ascent

- (Consider only equality constraints for ease of notation)
- We solve the dual problem using gradient ascent
- Assuming LD is differentiable, the gradient $\nabla LD(\lambda)$ can be evaluated as:
  1. Find $x^+ = \arg\min_x L(x, \lambda)$
  2. Compute $\nabla LD(\lambda) = h(x^+) = Ax^+ - b$

the residual for the (equality) constraint

Dual Ascent:

$$x^{k+1} = \arg\min_x L(x, \lambda^k)$$

$$\lambda^{k+1} = \lambda^k + s^k \left(Ax^{k+1} - b\right)$$

Step size > 0

Dual Ascent

- (Consider only equality constraints for ease of notation)
- We solve the dual problem using gradient ascent
- Assuming LD is differentiable, the gradient $\nabla LD(\lambda)$ can be evaluated as:

1. Find $x^+ = \arg\min_x L(x, \lambda)$
2. Compute $\nabla LD(\lambda) = h(x^+) = Ax^+ - b$

Dual Ascent:

$$\boxed{x^{k+1} = \arg\min_x L(x, \lambda^k)}$$

x-minimization step

$$\lambda^{k+1} = \lambda^k + s^k \left(Ax^{k+1} - b\right)$$

Dual Ascent

- (Consider only equality constraints for ease of notation)
- We solve the dual problem using gradient ascent
- Assuming LD is differentiable, the gradient $\nabla LD(\lambda)$ can be evaluated as:
  1. Find $x^+ = \arg\min_x L(x, \lambda)$
  2. Compute $\nabla LD(\lambda) = h(x^+) = Ax^+ - b$

Dual Ascent:

$$x^{k+1} = \arg\min_x L(x, \lambda^k)$$

$$\boxed{\lambda^{k+1} = \lambda^k + s^k \left(Ax^{k+1} - b\right)}$$

Dual variable update

Dual Decomposition

- Dual ascent can lead to a decentralized algorithm when *f* is separable (as in DCOPs)

$$\sum_{i=1}^{n} f_i(x_i) \;\; \text{s.t.} \;\; A_i x_i - b = 0 \;\; (i \in [n])$$

- The Lagrangian can be written as:

$$L(x, \lambda) = \sum_{i=1}^{n} L_i(x_i, \lambda) = \sum_{i=1}^{n} f_i(x_i) + \lambda^T (A_i x_i - b)$$

Dual Ascent:

$$x^{k+1} = \arg\min_x L(x, \lambda^k)$$

$$\lambda^{k+1} = \lambda^k + s^k \left(A x^{k+1} - b\right)$$

Dual Decomposition

- Dual ascent can lead to a decentralized algorithm when *f* is separable (as in DCOPs)

$$\sum_{i=1}^{n} f_i(x_i) \;\; \text{s.t.} \;\; A_i x_i - b = 0 \;\; (i \in [n])$$

- The Lagrangian can be written as:

$$L(x, \lambda) = \sum_{i=1}^{n} L_i(x_i, \lambda) = \sum_{i=1}^{n} f_i(x_i) + \lambda^T (A_i x_i - b)$$

Dual Decomposition:

$$\boxed{x^{k+1} = \arg\min_{x_i} L_i(x_i, \lambda^k), \quad i \in [n]}$$

$$\lambda^{k+1} = \lambda^k + s^k \left(\textstyle\sum_{i=1}^{n} A_i x_i^{k+1} - b\right)$$

Decentralized x-minimization step (in parallel)

Dual Decomposition

- Dual ascent can lead to a decentralized algorithm when *f* is separable (as in DCOPs)

$$\sum_{i=1}^{n} f_i(x_i) \;\; \text{s.t.} \;\; A_i x_i - b = 0 \;\; (i \in [n])$$

- The Lagrangian can be written as:

$$L(x, \lambda) = \sum_{i=1}^{n} L_i(x_i, \lambda) = \sum_{i=1}^{n} f_i(x_i) + \lambda^T (A_i x_i - b)$$

Dual Decomposition:

$$x^{k+1} = \arg\min_{x_i} L_i(x_i, \lambda^k), \quad i \in [n]$$

$$\boxed{\lambda^{k+1} = \lambda^k + s^k \left(\sum_{i=1}^{n} A_i x_i^{k+1} - b\right)}$$

Requires a "gather" step update and distribute the global λ variable

Dual Decomposition

- If the step size is well chosen and other assumptions hold, then x^k converges to an optimal point and λ^k to an optimal dual point.
- However, these assumptions do not hold in many applications.

Method of the Multipliers

- Used to robustly dual ascent
- Uses the *Augmented Lagrangian.*

Quadratic penalty makes the objective strongly convex

$$\min_{x \in \mathbb{R}^n} \; f(x) \quad \text{s.t.} \; Ax - b = 0$$

penalty

$$\longrightarrow \quad L_\rho(x, \lambda) = f(x) + \lambda^T(Ax - b) + \frac{\rho}{2}\|Ax - b\|_2^2$$

- Problem unchanged: has same local minima
- Advantage: The associated Lagrangian dual $\mathrm{LD}_\rho(\lambda) = \inf_x L_\rho(x, \lambda)$ can be shown to be differentiable.

Method of the Multipliers

- Advantage: The associated Lagrangian dual $\mathrm{LD}_\rho(\lambda) = \inf_x L_\rho(x, \lambda)$ can be shown to be differentiable.

$$x^{k+1} = \arg\min_x L_\rho(x, \lambda^k)$$
$$y^{k+1} = \lambda^k + \rho(Ax^{k+1} - b)$$

Specific step size

- Compared to decomposition, converges under milder assumptions (f can be non-differentiable, take on $+\infty$ values)
- However, the quadratic penalty destroys the splitting of the x-update, so cannot be decomposed

Alternating Method of the Multipliers (ADMM)

- Support decomposition.
- Consider a problem of the form (f, g, convex)

$$\begin{array}{ll} \text{minimize} & f(x) + g(z) \\ \text{subject to} & Ax + Bz = c \end{array}$$

- Two sets of variables with separate objective

$$L_p(x, z, \lambda) = f(x) + g(z) + \lambda^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax + Bz - c\|_2^2$$

Alternating Method of the Multipliers (ADMM)

- Support decomposition.
- Consider a problem of the form (f, g, convex)

$$\begin{array}{ll} \text{minimize} & f(x) + g(z) \\ \text{subject to} & Ax + Bz = c \end{array}$$

- Two sets of variables with separate objective

$$L_p(x, z, \lambda) = f(x) + g(z) + \lambda^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax + Bz - c\|_2^2$$

ADMM

$$x^{k+1} = \arg\min_x L_\rho(x, z^k, \lambda^k) \quad \text{x-minimization}$$

$$z^{k+1} = \arg\min_z L_\rho(x^{k+1}, z, \lambda^k) \quad \text{z-minimization}$$

$$\lambda^{k+1} = \lambda^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \quad \text{dual update}$$

Alternating Method of the Multipliers
(ADMM)

- If we minimized over x and z jointly, reduces to method of the multipliers.
- We can decompose because we minimize over x with fixed z, and vice-versa.

ADMM

$$x^{k+1} = \arg\min_x L_\rho(x, z^k, \lambda^k) \quad \text{x-minimization}$$

$$z^{k+1} = \arg\min_z L_\rho(x^{k+1}, z, \lambda^k) \quad \text{z-minimization}$$

$$\lambda^{k+1} = \lambda^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \quad \text{dual update}$$

Convergence
(ADMM)

- Mild assumptions
  - f, g, convex, closed, proper

- Then ADMM converges:
  - Iterates approach feasibility $Ax^k + Bz^k - c \to 0$
  - Objective approaches optimal value: $f(x^k) + g(z^k) \to$ OPT

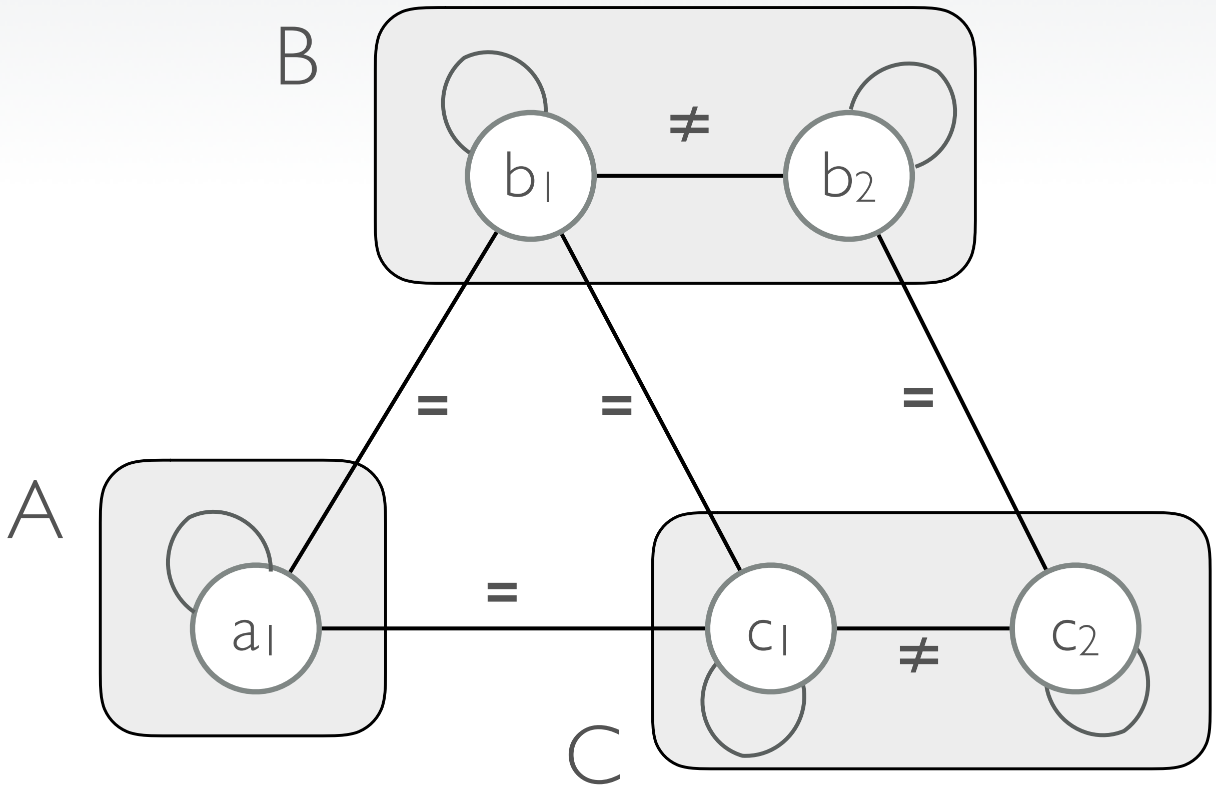
Applications

AAAI-20 Tutorial on
Multi-Agent Distributed Constrained Optimization

DCOP APPLICATIONS

- Scheduling Problems
  - Taking DCOP to the Real World: Efficient Complete Solutions for Distributed Multi-Event Scheduling. AAMAS 2004
- Radio Frequency Allocation Problems
  - Improving DPOP with Branch Consistency for Solving Distributed Constraint Optimization Problems. CP 2014
- Sensor Networks
  - Preprocessing techniques for accelerating the DCOP algorithm ADOPT. AAMAS 2005
- Home Automation
  - A Multiagent System Approach to Scheduling Devices in Smart Homes. AAMAS 2017, IJCAI 2016
- Traffic Light Synchronization
  - Evaluating the performance of DCOP algorithms in a real world, dynamic problem. AAMAS 2008
- Disaster Evacuation
  - Disaster Evacuation Support. AAAI 2007; JAIR 2017
- Combinatorial Auction Winner Determination
  - H-DPOP: Using Hard Constraints for Search Space Pruning in DCOP. AAAI 2008

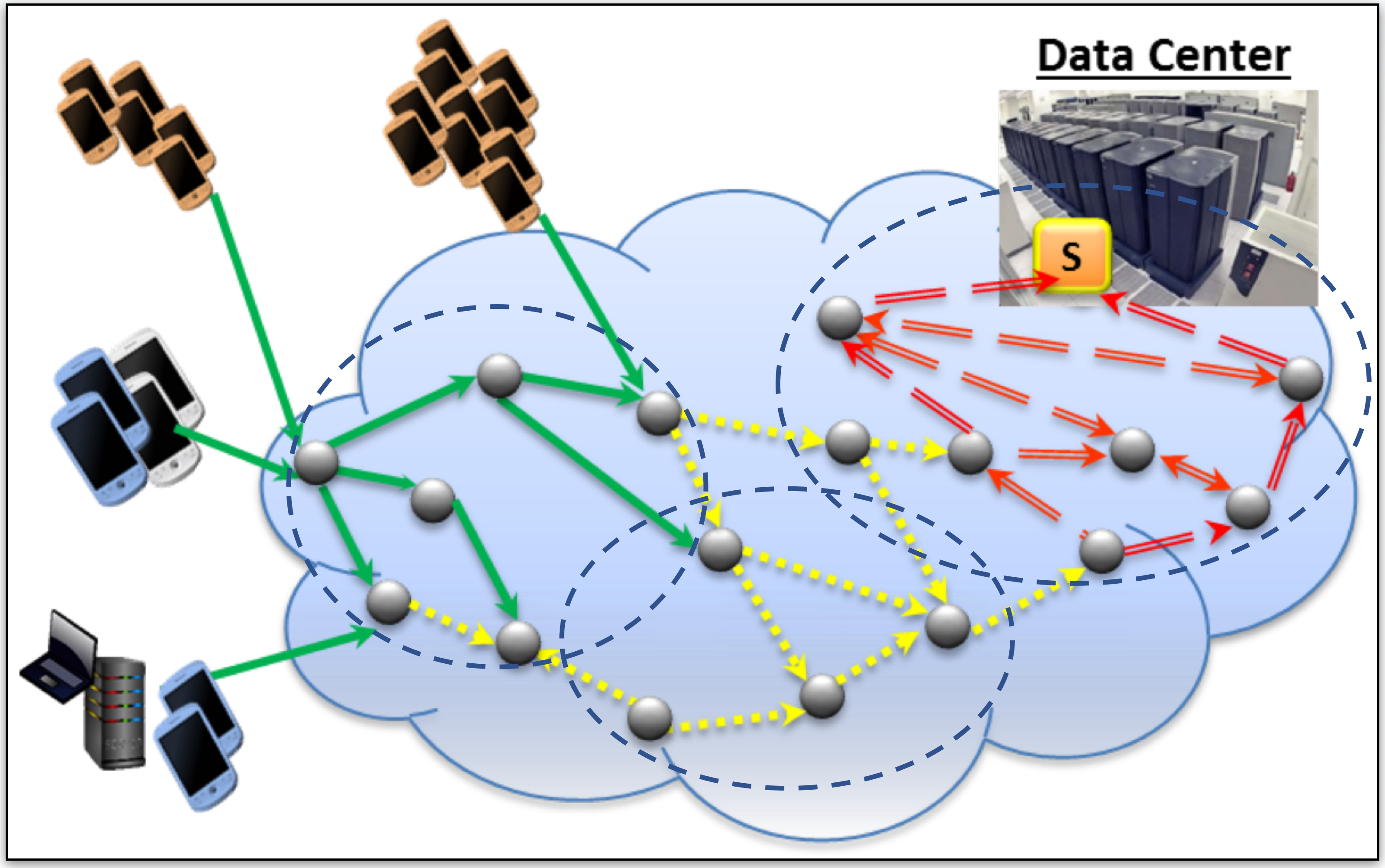
Meeting Scheduling



- Values: time slots to hold the meetings
- All agents participating in a meeting must meet at the same time
- All meetings of an agent must occur at different times

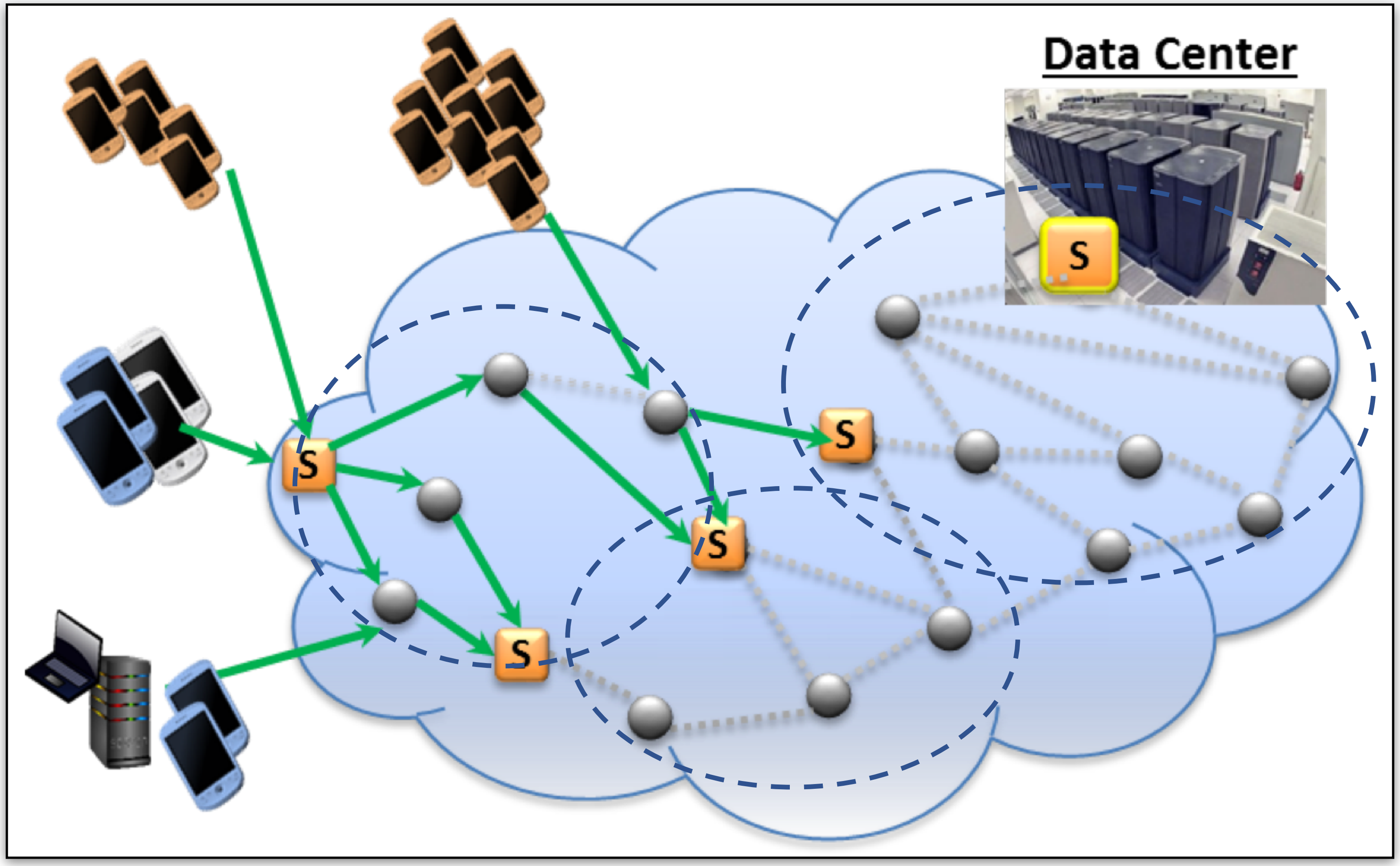
Rajiv T. Maheswaran, Milind Tambe, Emma Bowring, Jonathan P. Pearce, Pradeep Varakantham: Taking DCOP to the Real World: Efficient Complete Solutions for Distributed Multi-Event Scheduling. AAMAS 2004: 310-317

Edge Computing



Khoi D. Hoang, Christabel Wayllace, William Yeoh, Jacob Beal, Soura Dasgupta, Yuanqiu Mo, Aaron Paulos, Jon Schewe: New Distributed Constraint Reasoning Algorithms for Load Balancing in Edge Computing. PRIMA 2019: 69-86

Edge Computing



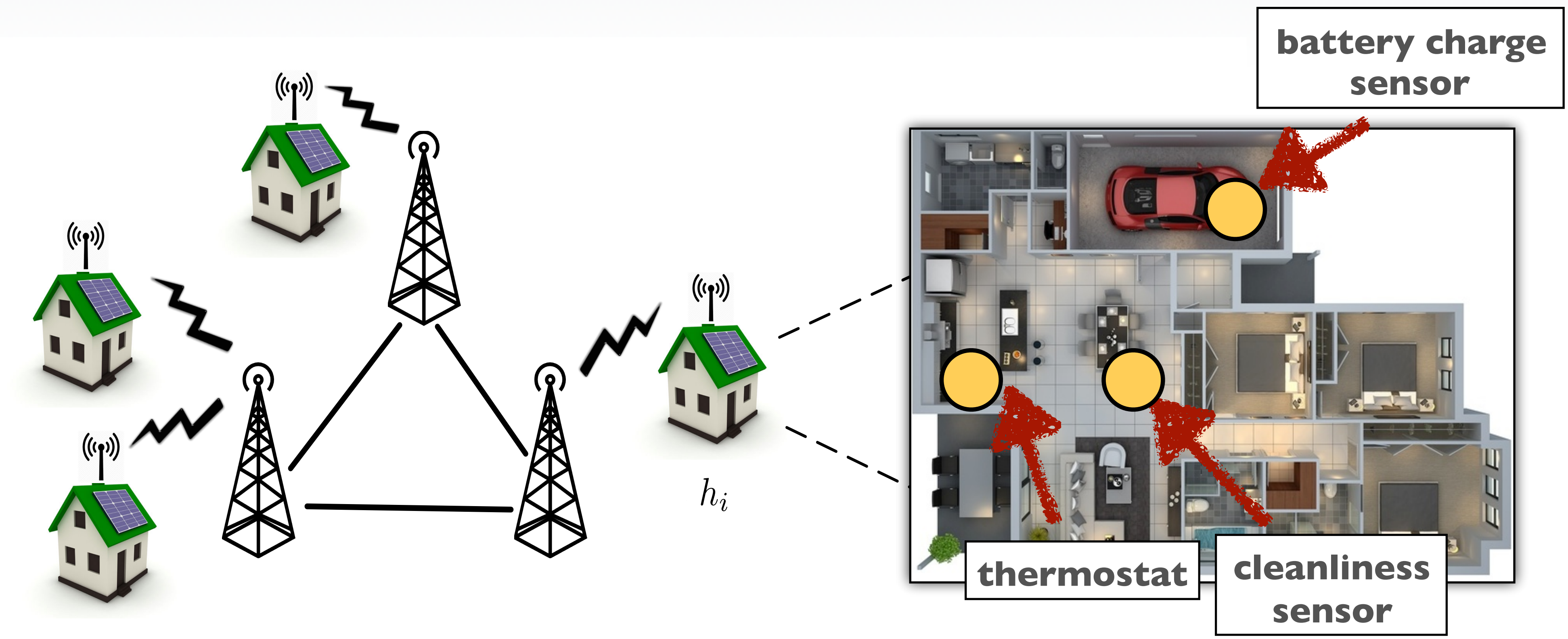
Khoi D. Hoang, Christabel Wayllace, William Yeoh, Jacob Beal, Soura Dasgupta, Yuanqiu Mo, Aaron Paulos, Jon Schewe: New Distributed Constraint Reasoning Algorithms for Load Balancing in Edge Computing. PRIMA 2019: 69-86

Edge Computing

- Agents: a_v for each vertex $v \in V$
- Variables: $v_{(v, s, c)}$
  - Denotes amount of load to serve for service s by client c on vertex v
  - Each controlled by agent a_v
- Domain: $0 \leq D_{(v, s, c)} \leq cap(v)$
- Constraints:
  - $\sum_{(s \in S, c \in C)} v_{(v, s, c)} \leq cap(v)$
  - $\sum_{(v \in V\ s \in S, c \in C)} v_{(v, s, c)} \geq \sum_{(v \in V\ s \in S, c \in C)} load(v,s,c)$
- Maximize: $\sum_{(v \in V\ s \in S, c \in C)} v_{(v, s, c)} / dist(v,c)$

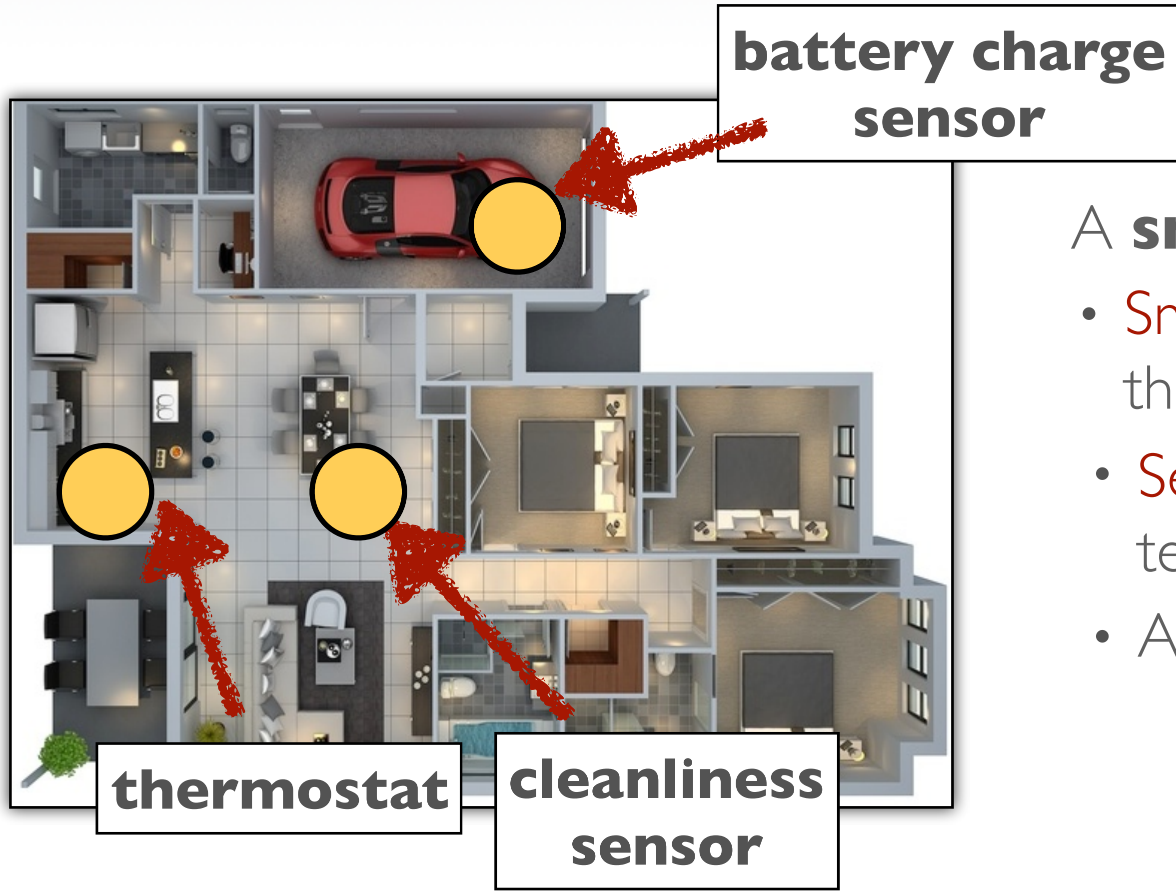
Khoi D. Hoang, Christabel Wayllace, William Yeoh, Jacob Beal, Soura Dasgupta, Yuanqiu Mo, Aaron Paulos, Jon Schewe: New Distributed Constraint Reasoning Algorithms for Load Balancing in Edge Computing. PRIMA 2019: 69-86

Smart Homes Device Scheduling



Ferdinando Fioretto, William Yeoh, Enrico Pontelli. "A Multiagent System Approach to Scheduling Devices in Smart Homes". AAMAS, 2017.

Smart Homes Device Scheduling

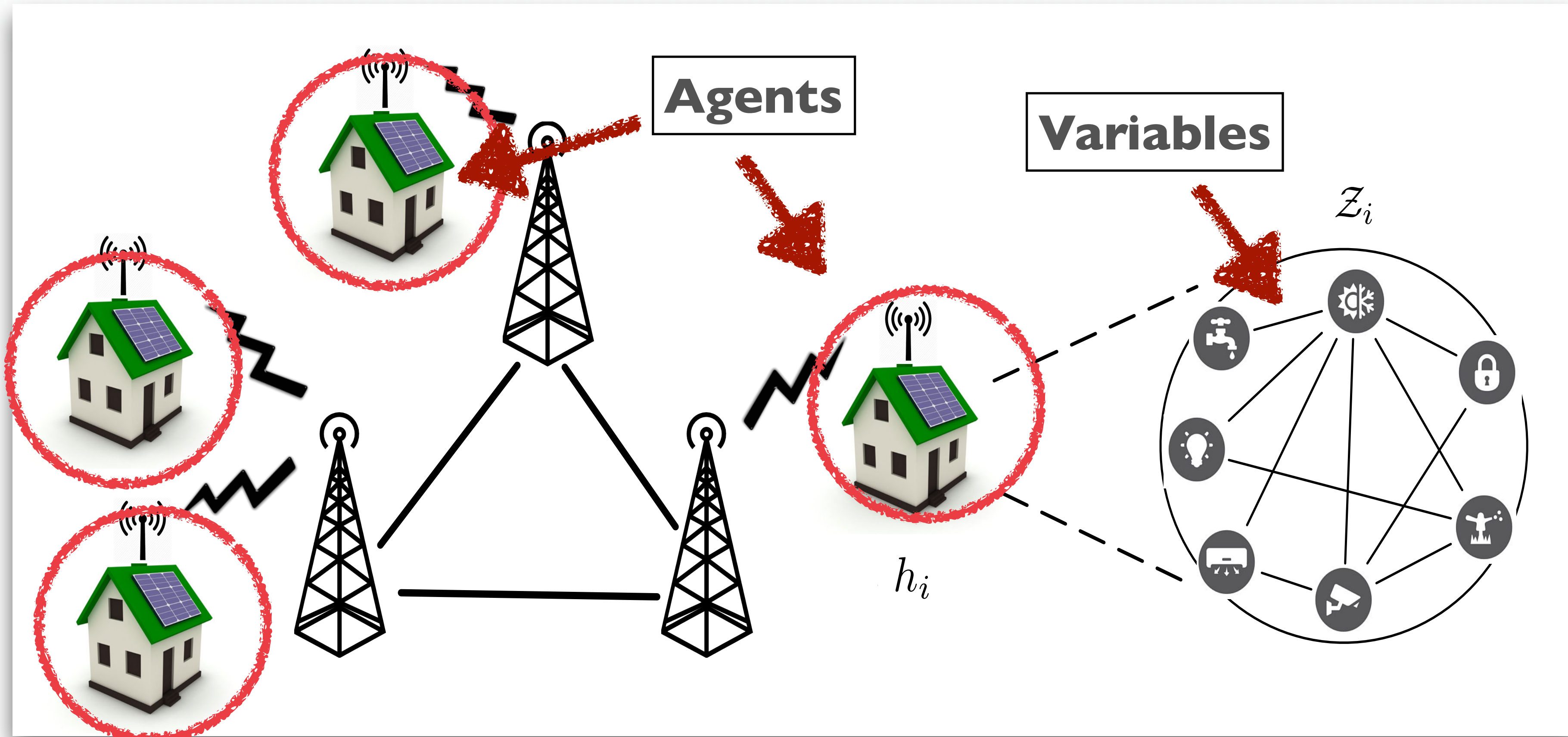


A **smart home** has:

- Smart devices (roomba, HVAC) that it can control
- Sensors (cleanliness, temperature)
- A set of locations

Ferdinando Fioretto, William Yeoh, Enrico Pontelli. "A Multiagent System Approach to Scheduling Devices in Smart Homes". AAMAS, 2017.

Smart Homes
Device Scheduling



Ferdinando Fioretto, William Yeoh, Enrico Pontelli. "A Multiagent System Approach to Scheduling Devices in Smart Homes". AAMAS, 2017.

Challenges and Open Questions

AAAI-20 Tutorial on
Multi-Agent Distributed Constrained Optimization

MAS Coordination

- Decentralized coordination in a MAS is expensive

- William Yeoh, Pradeep Varakantham, Xiaoxun Sun, and Sven Koenig. "Incremental DCOP Search Algorithms for Solving Dynamic DCOP Problems." In Proceedings of the International Conference on Intelligent Agent Technology (IAT), pages 257-263, 2015.
- Duc Thien Nguyen, William Yeoh, and Hoong Chuin Lau. "Distributed Gibbs: A Memory-Bounded Sampling-Based DCOP Algorithm." In Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pages 167-174, 2013.

MAS Coordination

- Decentralized coordination in a MAS is expensive
  - Can we study various tradeoff (solution quality vs. runtime vs. communication time) to improve coordination?
  - Can we use sampling methods to develop new, efficient, anytime, DCOP incomplete algorithms?

- William Yeoh, Pradeep Varakantham, Xiaoxun Sun, and Sven Koenig. "Incremental DCOP Search Algorithms for Solving Dynamic DCOP Problems." In Proceedings of the International Conference on Intelligent Agent Technology (IAT), pages 257-263, 2015.
- Duc Thien Nguyen, William Yeoh, and Hoong Chuin Lau. "Distributed Gibbs: A Memory-Bounded Sampling-Based DCOP Algorithm." In Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pages 167-174, 2013.

Dynamic Environment

- Interaction in a dynamic environment is required to be robust to several changes

- R. Mailler, H. Zheng, and A. Ridgway. 2017. Dynamic, distributed constraint solving and thermodynamic theory. Auton Agent Multi-Agent Syst (2017).
- Zhang, C., & Lesser, V. (2013). Coordinating multi-agent reinforcement learning with limited communication. In Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 1101–1108.

Dynamic Environment

- Interaction in a dynamic environment is required to be robust to several changes
  - How do agents respond to dynamic changes?
  - Can we study adaptive algorithms so that the MAS interaction is resilient and adaptive to changes in the communication layer, the underlying constraint graph, etc.?

- R. Mailler, H. Zheng, and A. Ridgway. 2017. Dynamic, distributed constraint solving and thermodynamic theory. Auton Agent Multi-Agent Syst (2017).
- Zhang, C., & Lesser, V. (2013). Coordinating multi-agent reinforcement learning with limited communication. In Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 1101–1108.

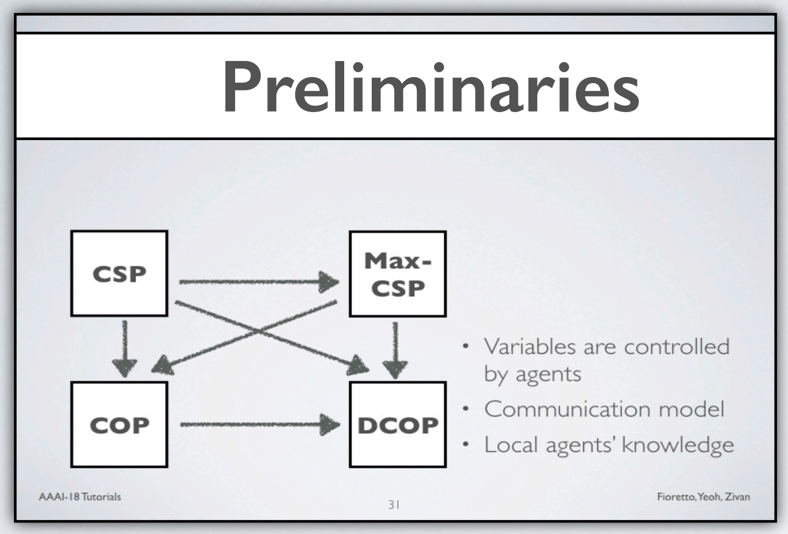
Agent Preferences

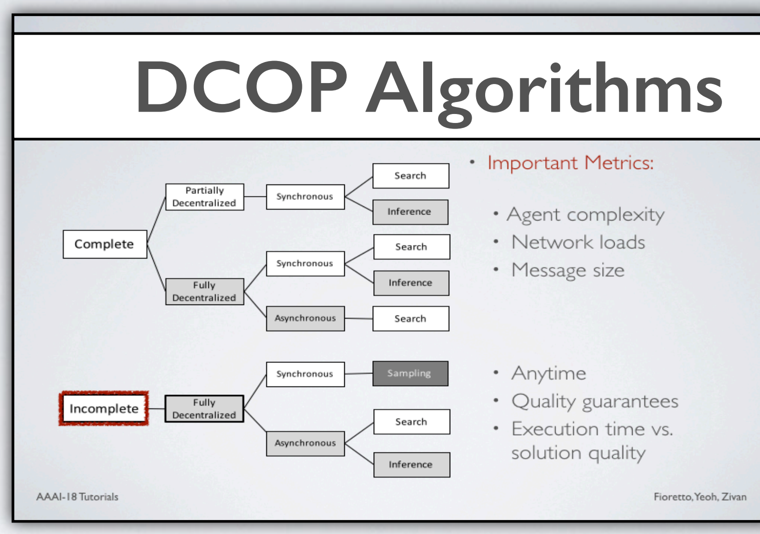
- How to model, learn, and update agent preferences?

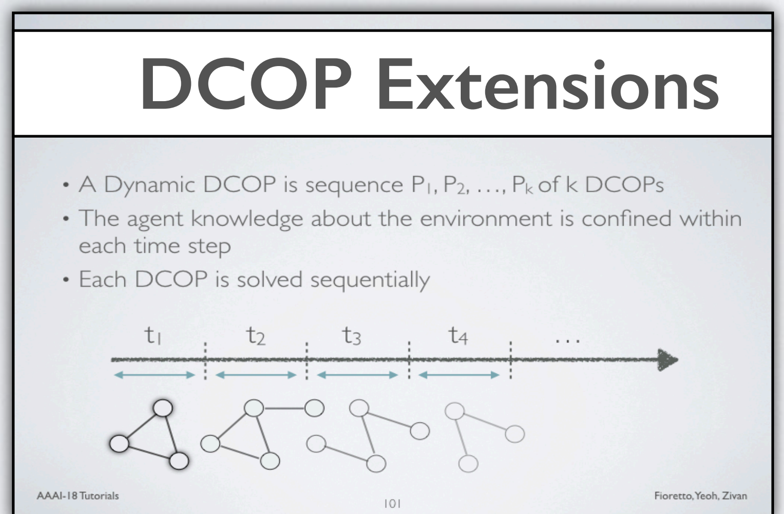
Agent Preferences

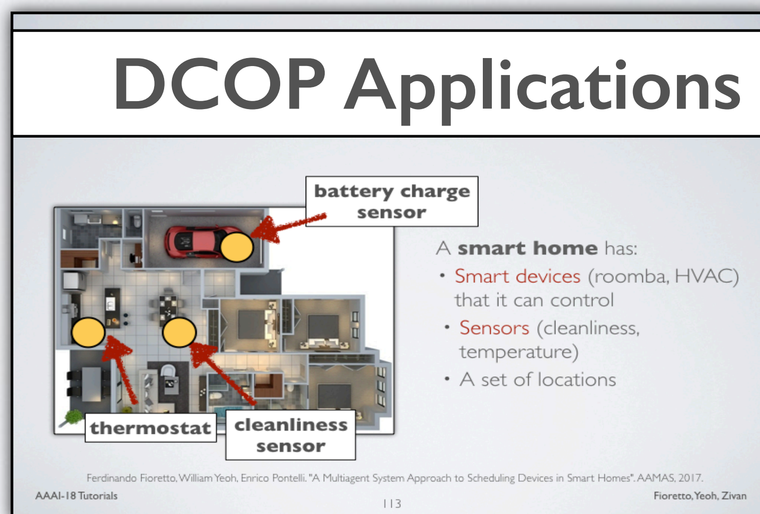
- How to model, learn, and update agent preferences?
  - Agent's preferences are assumed to be available. This is not always feasible. How to efficiently elicit agents' preferences?
  - When full elicitation is not possible, how to adaptively learn the preference of an agent?

- Atena M. Tabakhi, Tiep Le, Ferdinando Fioretto, and William Yeoh. "Preference Elicitation for DCOPs." In Proceedings of the International Conference on Principles and Practice of Constraint Programming (CP), pages 278-296, 2017









Thank You!

Ferdinando Fioretto & William Yeoh