Infinite-Horizon Proactive Dynamic DCOPs

Khoi Hoang
Ping Hou
William Yeoh

Ferdinando Fioretto

Roie Zivan

Makoto Yokoo

Outline

- Distributed Constraint Optimization Problems
- **Dynamic** DCOPs
- **Proactive** Dynamic DCOPs
- **Infinite-Horizon** Proactive Dynamic DCOPs*
- Overview and Details

Outline

- Distributed Constraint Optimization Problems
- **Dynamic** DCOPs
- **Proactive** Dynamic DCOPs
- **Infinite-Horizon** Proactive Dynamic DCOPs*
- Overview and Details

Outline

- Distributed Constraint Optimization Problems
- **Dynamic** DCOPs
- **Proactive** Dynamic DCOPs
- **Infinite-Horizon** Proactive Dynamic DCOPs*
- Overview and Details

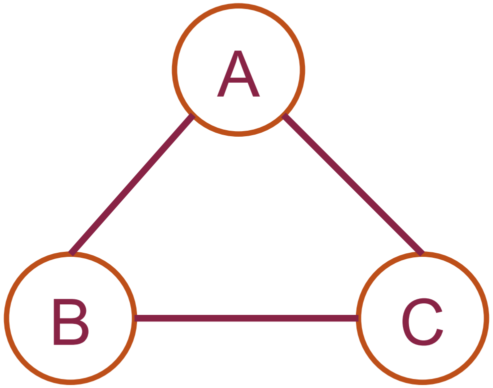
Outline

- Distributed Constraint Optimization Problems
- **Dynamic** DCOPs
- **Proactive** Dynamic DCOPs
- **Infinite-Horizon** Proactive Dynamic DCOPs*
- Overview and Details

Outline

- Distributed Constraint Optimization Problems
- **Dynamic** DCOPs
- **Proactive** Dynamic DCOPs
- **Infinite-Horizon** Proactive Dynamic DCOPs*
- Overview and Details

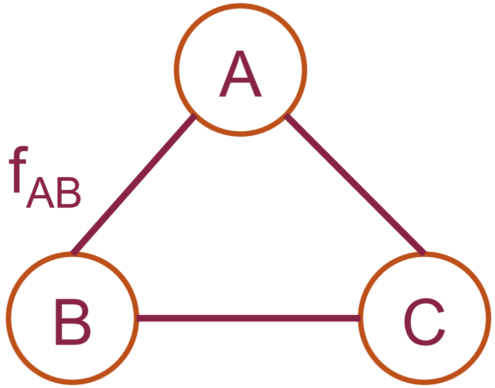
Distributed Constraint Optimization Problems



[1] Modi et al., ADOPT: Asynchronous Distributed Constraint Optimization with Quality Guarantees, 2005

Distributed Constraint Optimization Problems

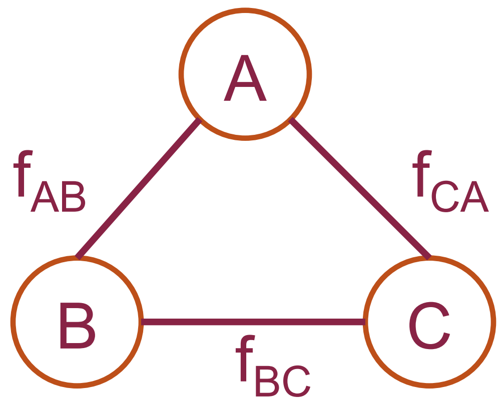
x_A	x_B	$f_{AB}(x_A,x_B)$
0	0	5
0	1	10
…	…	…



[1] Modi et al., ADOPT: Asynchronous Distributed Constraint Optimization with Quality Guarantees, 2005

Distributed Constraint Optimization Problems

x_A	x_B	$f_{AB}(x_A,x_B)$
0	0	5
0	1	10
…	…	…

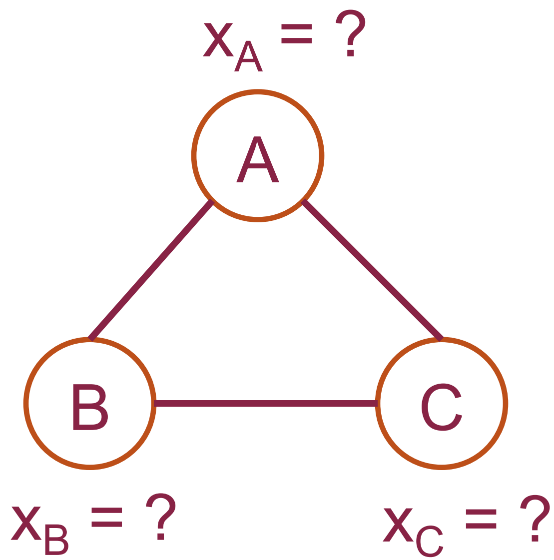


x_C	x_A	$f_{CA}(x_C,x_A)$
0	0	7
0	1	4
…	…	…

x_B	x_C	$f_{BC}(x_B,x_C)$
0	0	3
0	1	12
…	…	…

[1] Modi et al., ADOPT: Asynchronous Distributed Constraint Optimization with Quality Guarantees, 2005

Distributed Constraint Optimization Problems



$\boxed{\text{Maximize } f_{AB} + f_{BC} + f_{CA}}$

[1] Modi et al., ADOPT: Asynchronous Distributed Constraint Optimization with Quality Guarantees, 2005

Distributed Constraint Optimization Problems

- Meeting scheduling problems
- Smart devices scheduling
- Resource allocation
- Sensor network

Limitations

- DCOPs
  - Static problem
  - Not consider possible changes

Limitations

- DCOPs
  - Static problem
  - Not consider possible changes
- Dynamic DCOPs
  - Reacting to changes of the problem

[2] R. Lass et al., Dynamic distributed constraint reasoning, 2005
[3] A. Petcu and B. Faltings, Superstabilizing, fault-containing multiagent combinatorial optimization, 2005

Dynamic DCOPs

- DCOPs
  - Static problem
  - Not consider possible changes
- Dynamic DCOPs
  - Reacting to changes of the problem

New Mexico State University
All About Discovery!
nmsu.edu

[2] R. Lass et al., Dynamic distributed constraint reasoning, 2005
[3] A. Petcu and B. Faltings, Superstabilizing, fault-containing multiagent combinatorial optimization, 2005

Dynamic DCOPs

- DCOPs
  - Static problem
  - Not consider possible changes
- Dynamic DCOPs
  - Reacting to changes of the problem

[2] R. Lass et al., Dynamic distributed constraint reasoning, 2005
[3] A. Petcu and B. Faltings, Superstabilizing, fault-containing multiagent combinatorial optimization, 2005

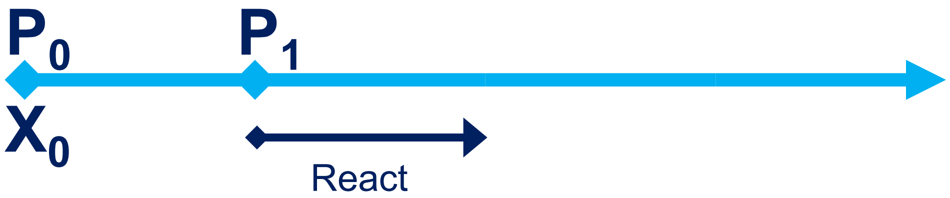
Dynamic DCOPs

- DCOPs
  - Static problem
  - Not consider possible changes
- Dynamic DCOPs
  - Reacting to changes of the problem

[2] R. Lass et al., Dynamic distributed constraint reasoning, 2005
[3] A. Petcu and B. Faltings, Superstabilizing, fault-containing multiagent combinatorial optimization, 2005

Dynamic DCOPs

- DCOPs
  - Static problem
  - Not consider possible changes
- Dynamic DCOPs
  - Reacting to changes of the problem



[2] R. Lass et al., Dynamic distributed constraint reasoning, 2005
[3] A. Petcu and B. Faltings, Superstabilizing, fault-containing multiagent combinatorial optimization, 2005

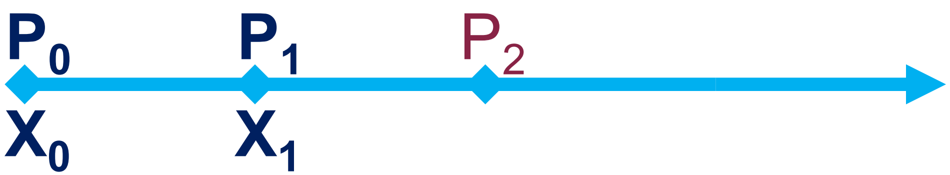
Dynamic DCOPs

- DCOPs
  - Static problem
  - Not consider possible changes
- Dynamic DCOPs
  - Reacting to changes of the problem

[2] R. Lass et al., Dynamic distributed constraint reasoning, 2005
[3] A. Petcu and B. Faltings, Superstabilizing, fault-containing multiagent combinatorial optimization, 2005

Dynamic DCOPs

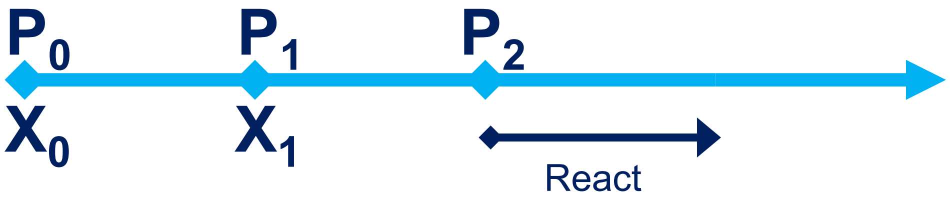
- DCOPs
  – Static problem
  – Not consider possible changes
- Dynamic DCOPs
  – Reacting to changes of the problem



[2] R. Lass et al., Dynamic distributed constraint reasoning, 2005
[3] A. Petcu and B. Faltings, Superstabilizing, fault-containing multiagent combinatorial optimization, 2005

Dynamic DCOPs

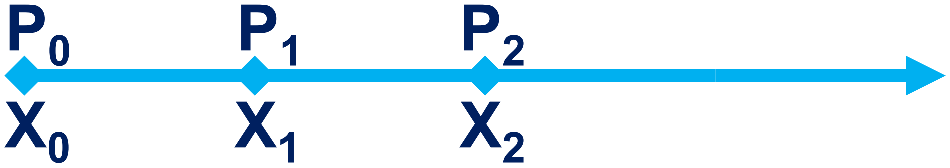
- DCOPs
  - Static problem
  - Not consider possible changes
- Dynamic DCOPs
  - Reacting to changes of the problem



[2] R. Lass et al., Dynamic distributed constraint reasoning, 2005
[3] A. Petcu and B. Faltings, Superstabilizing, fault-containing multiagent combinatorial optimization, 2005

Dynamic DCOPs

- DCOPs
  - Static problem
  - Not consider possible changes
- Dynamic DCOPs
  - Reacting to changes of the problem



[2] R. Lass et al., Dynamic distributed constraint reasoning, 2005
[3] A. Petcu and B. Faltings, Superstabilizing, fault-containing multiagent combinatorial optimization, 2005

Limitations

- Dynamic DCOPs
  - Not take advantage of possible changes
  - Good for current, bad for future (myopic solutions)

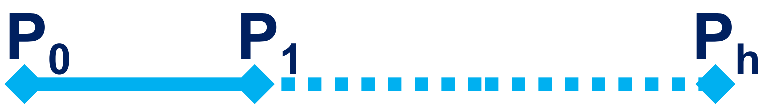
[2] R. Lass et al., Dynamic distributed constraint reasoning, 2005
[3] A. Petcu and B. Faltings, Superstabilizing, fault-containing multiagent combinatorial optimization, 2005

Limitations

- Dynamic DCOPs
  - Not take advantage of possible changes
  - Good for current, bad for future (myopic solutions)
- How about if we know
  - How often the problems change
  - Knowledge about possible changes

Proactive Dynamic DCOPs

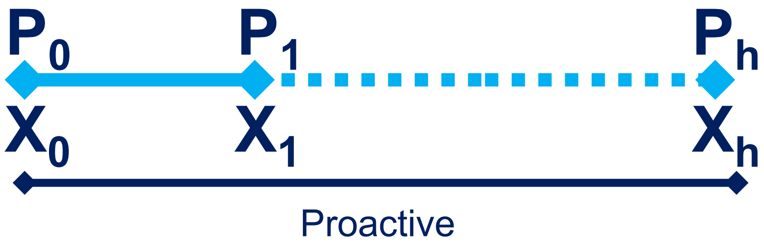
- Knowledge about changes of random events
  - Initial distribution and transition function
- Solve all the problems beforehand up to horizon h
- Keep the solution at time step h



[4] Hoang et al., Proactive Dynamic Distributed Constraint Optimization, 2016

Proactive Dynamic DCOPs

- Knowledge about changes of random events
  - Initial distribution and transition function
- Solve all the problems beforehand up to horizon h
- Keep the solution at time step h



[4] Hoang et al., Proactive Dynamic Distributed Constraint Optimization, 2016

Limitations

Is this solution optimal from h onwards???

[4] Hoang et al., Proactive Dynamic Distributed Constraint Optimization, 2016

Key contributions

- Infinite-Horizon Proactive Dynamic DCOPs
  - Optimal solution from h onwards
  - Based on converged distribution at h*
  - Proactive vs. Reactive dynamic DCOP algorithms (first time!!!)

Key contributions

- Infinite-Horizon Proactive Dynamic DCOPs
  - Optimal solution from h onwards
  - Based on converged distribution at h*
  - Proactive vs. Reactive dynamic DCOP algorithms (first time!!!)

Key contributions

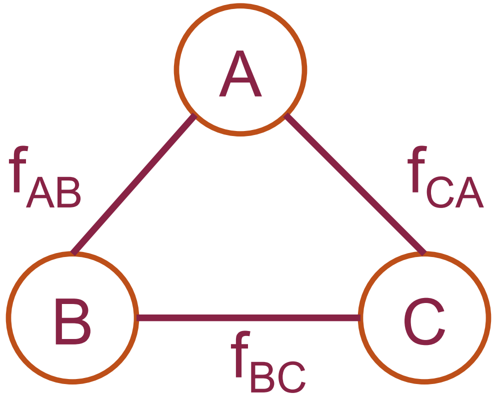
- Infinite-Horizon Proactive Dynamic DCOPs
  - Optimal solution from h onwards
  - Based on converged distribution at h*
  - Proactive vs. Reactive dynamic DCOP algorithms (first time!!!)

Content

- ➢ Distributed Constraint Optimization Problem
- Proactive Dynamic DCOPs
- Infinite-Horizon Proactive Dynamic DCOPs
- Algorithms
- Experiments
- Conclusions

Distributed Constraint Optimization Problems

x_A	x_B	$f_{AB}(x_A,x_B)$
0	0	5
0	1	10
…	…	…

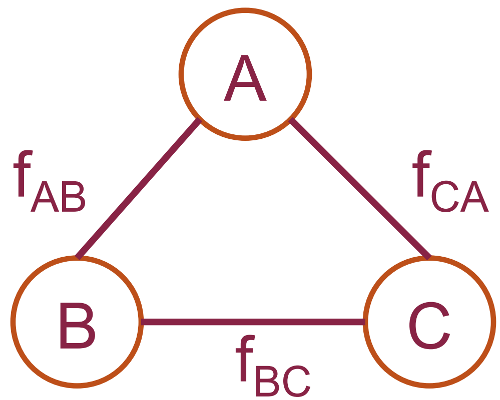


x_C	x_A	$f_{CA}(x_C,x_A)$
0	0	7
0	1	4
…	…	…

x_B	x_C	$f_{BC}(x_B,x_C)$
0	0	3
0	1	12
…	…	…

Distributed Constraint Optimization Problems

x_A	x_B	$f_{AB}(x_A,x_B)$
0	0	5
0	1	10
…	…	…

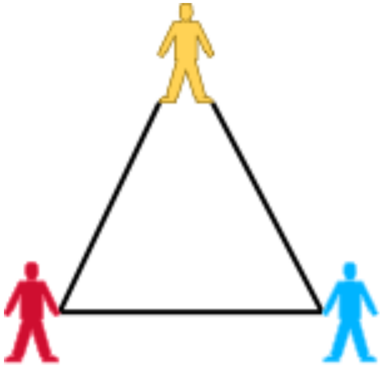


x_C	x_A	$f_{CA}(x_C,x_A)$
0	0	7
0	1	4
…	…	…

x_B	x_C	$f_{BC}(x_B,x_C)$
…	…	…

Maximize $f_{AB} + f_{BC} + f_{CA}$

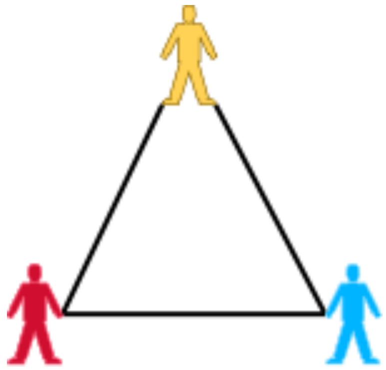
Distributed Meeting Scheduling Problem



[5] Maheswaran et al., Taking DCOP to the real world: efficient complete solutions for distributed event scheduling, 2004.

Distributed Meeting Scheduling Problem

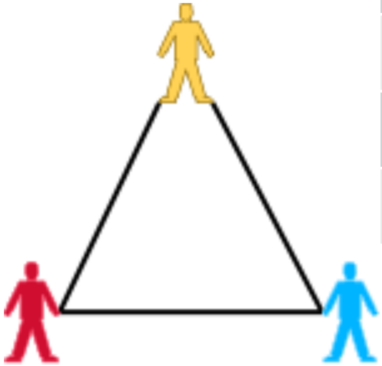
Person A	Utility
8:00	2
9:00	5
…	…
16:00	10



[5] Maheswaran et al., Taking DCOP to the real world: efficient complete solutions for distributed event scheduling, 2004.

Distributed Meeting Scheduling Problem

Person A	Utility
8:00	2
9:00	5
…	…
16:00	10



Person A	Person B	Utility
8:00	8:00	0
8:00	9:00	- infinity
…	…	…
16:00	16:00	0

$$\text{Maximize } f_A + f_B + f_C + f_{AB} + f_{BC} + f_{CA}$$

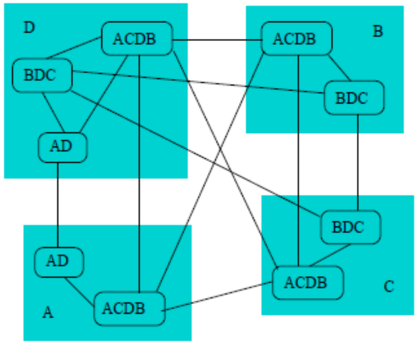
[5] Maheswaran et al., Taking DCOP to the real world: efficient complete solutions for distributed event scheduling, 2004.

Distributed Constraint Optimization Problem (DCOP)

DCOP is a tuple <A, X, D, F>

- $A = \{a_1, a_2, \ldots, a_n\}$
- $X = \{x_1, x_2, \ldots, x_m\}$
- $D = \{D_1, D_2, \ldots, D_m\}$
- $F = \{f_1, f_2, \ldots, f_l\}$
- $F(\sigma) = \Sigma f_i$
- $\sigma_{max} = \text{argmax } F(\sigma)$

Person A	Utility
8:00	2
9:00	5
…	…
16:00	10



New Mexico State University
All About Discovery!
nmsu.edu

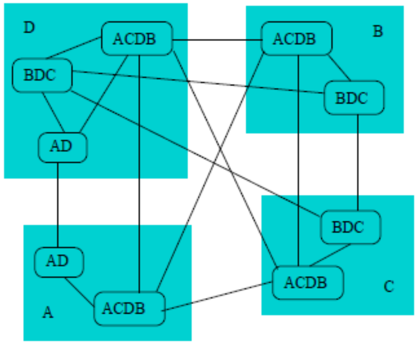
[1] Modi et al., ADOPT: Asynchronous Distributed Constraint Optimization with Quality Guarantees, 2005

Distributed Constraint Optimization Problem (DCOP)

DCOP is a tuple <A, X, D, F>

- $A = \{a_1, a_2, \ldots, a_n\}$
- $X = \{x_1, x_2, \ldots, x_m\}$
- $D = \{D_1, D_2, \ldots, D_m\}$
- $F = \{f_1, f_2, \ldots, f_l\}$
- $F(\sigma) = \Sigma f_i$
- $\sigma_{max} = \text{argmax } F(\sigma)$

Person A	Utility
8:00	2
9:00	5
…	…
16:00	10



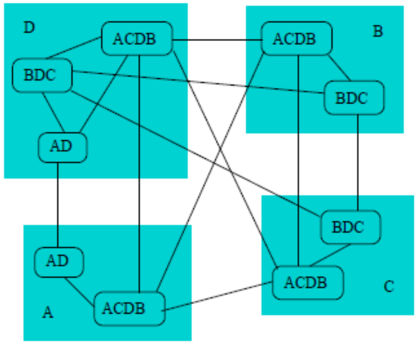
[1] Modi et al., ADOPT: Asynchronous Distributed Constraint Optimization with Quality Guarantees, 2005

Distributed Constraint Optimization Problem (DCOP)

DCOP is a tuple <A, X, D, F>

- $A = \{a_1, a_2, \ldots, a_n\}$
- $X = \{x_1, x_2, \ldots, x_m\}$
- $D = \{D_1, D_2, \ldots, D_m\}$
- $F = \{f_1, f_2, \ldots, f_l\}$
- $F(\sigma) = \Sigma f_i$
- $\sigma_{max} = \text{argmax } F(\sigma)$

Person A	Utility
8:00	2
9:00	5
…	…
16:00	10



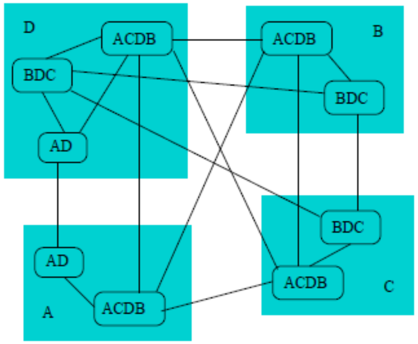
[1] Modi et al., ADOPT: Asynchronous Distributed Constraint Optimization with Quality Guarantees, 2005

Distributed Constraint Optimization Problem (DCOP)

DCOP is a tuple <A, X, D, F>

- $A = \{a_1, a_2, \ldots, a_n\}$
- $X = \{x_1, x_2, \ldots, x_m\}$
- $D = \{D_1, D_2, \ldots, D_m\}$
- $F = \{f_1, f_2, \ldots, f_l\}$
- $F(\sigma) = \Sigma f_i$
- $\sigma_{max} = \text{argmax } F(\sigma)$

Person A	Utility
8:00	2
9:00	5
…	…
16:00	10



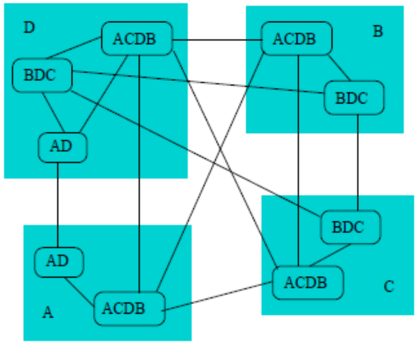
[1] Modi et al., ADOPT: Asynchronous Distributed Constraint Optimization with Quality Guarantees, 2005

Distributed Constraint Optimization Problem (DCOP)

DCOP is a tuple <A, X, D, F>

- $A = \{a_1, a_2, \ldots, a_n\}$
- $X = \{x_1, x_2, \ldots, x_m\}$
- $D = \{D_1, D_2, \ldots, D_m\}$
- $F = \{f_1, f_2, \ldots, f_l\}$
- $F(\sigma) = \Sigma f_i$
- $\sigma_{max} = \text{argmax } F(\sigma)$

Person A	Utility
8:00	2
9:00	5
…	…
16:00	10



[1] Modi et al., ADOPT: Asynchronous Distributed Constraint Optimization with Quality Guarantees, 2005

Content

- Distributed Constraint Optimization Problem
- ➢ Proactive Dynamic DCOPs
- Infinite-Horizon Proactive Dynamic DCOP
- Algorithms
- Experiments
- Conclusions

Proactive Dynamic DCOPs

- Random variables
  - Initial distribution
  - Transition function

Week 0	Raining
	8:00

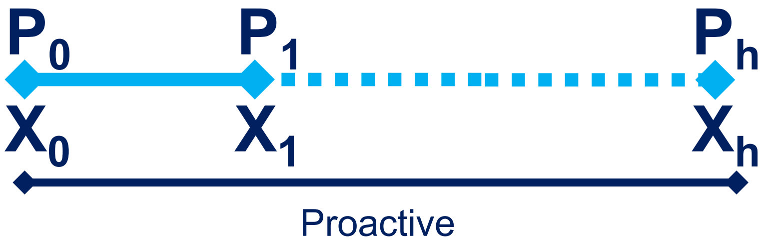
→

Week 1	Raining
	10:00

$$P = \begin{bmatrix} 0.9 & 0.075 & 0.025 \\ 0.15 & 0.8 & 0.05 \\ 0.25 & 0.25 & 0.5 \end{bmatrix}$$

[4] Hoang et al., Proactive Dynamic Distributed Constraint Optimization, 2016

Proactive Dynamic DCOPs

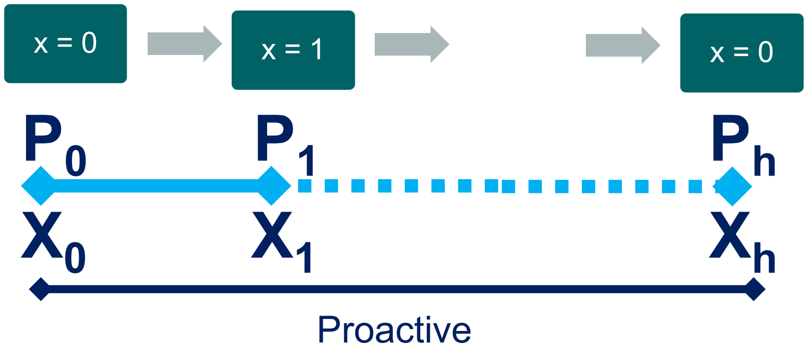


- Constraints with random variables
- Proactive: solve the whole problems beforehand
- Keep the solution at h onwards

[4] Hoang et al., Proactive Dynamic Distributed Constraint Optimization, 2016

Proactive Dynamic DCOPs

- $Y = \{y_1, y_2, \ldots, y_m\}$
  - Ω: event space
  - p^0: initial distribution
  - T: transition function
- c: switching cost
- h: horizon



[4] Hoang et al., Proactive Dynamic Distributed Constraint Optimization, 2016

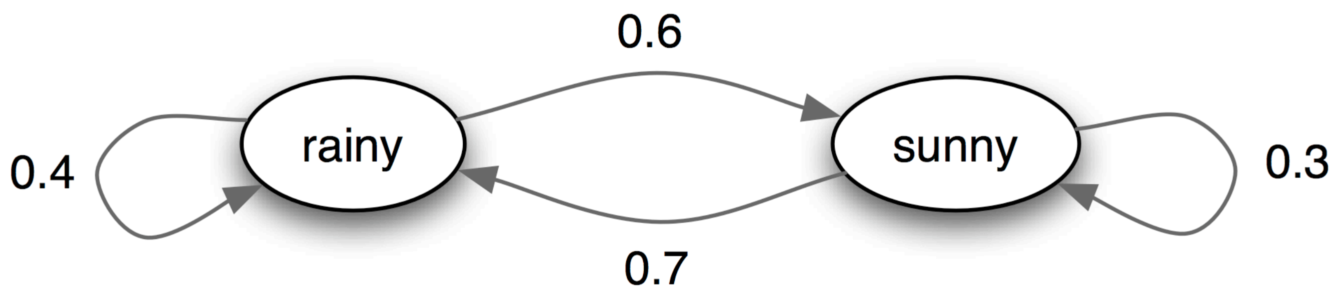
Content

- Distributed Constraint Optimization Problem
- Proactive Dynamic DCOPs
- ➢ Infinite-Horizon Proactive Dynamic DCOP
- Algorithms
- Experiments
- Conclusions

Limitations

Is this solution optimal from h onwards???

Infinite-Horizon Proactive Dynamic DCOPs



- Each random variable => Markov chain
- Optimal solutions from h onwards
  - ➢ Markov chain convergence

Infinite-Horizon Proactive Dynamic DCOPs

- Under some specific conditions:
  - Markov chains converge at h*
  - Solve the problem at h with converged distribution

Content

- Distributed Constraint Optimization Problem
- Proactive Dynamic DCOPs
- Infinite-Horizon Proactive Dynamic DCOP
- ➢ Algorithms
- Experiments
- Conclusions

Algorithms

- Preprocessing
  - Eliminate random variables
  - Calculate expected utility
  - Regular DCOPs at every time step
- FORWARD
- BACKWARD

Preprocessing (cont.)

- Constraints with random variables

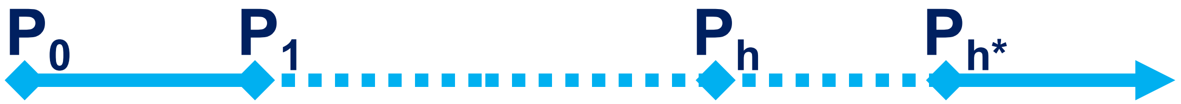
x	y	
0	0	a
0	1	b
1	0	c
1	1	d

x	ts = k
0	a*prob(y=0) + b*prob(y=1)
1	c*prob(y=0) + d*prob(y=1)

- Regular DCOPs at every time step

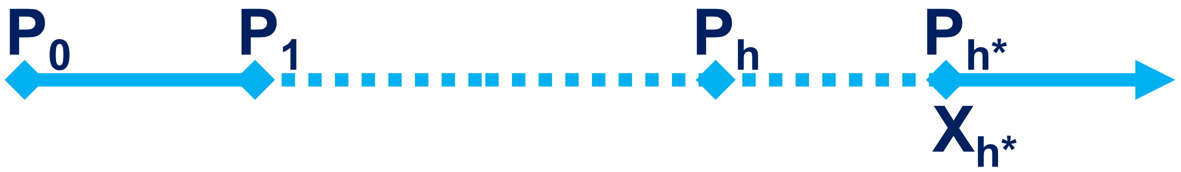
FORWARD

- Solve problem at h with converged distribution
- Solve from P_0 forward
- Online – Offline algorithm



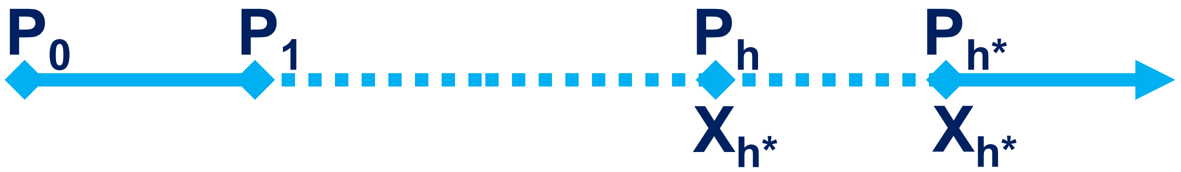
FORWARD

- Solve problem at h with converged distribution
- Solve from P_0 forward
- Online – Offline algorithm



FORWARD

- Solve problem at h with converged distribution
- Solve from P_0 forward
- Online – Offline algorithm

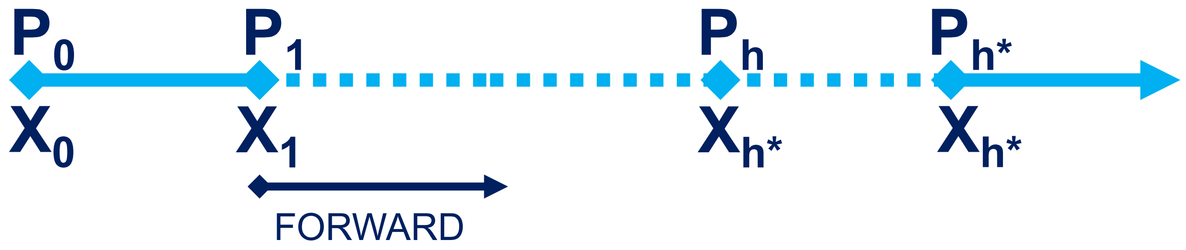


FORWARD

- Solve problem at h with converged distribution
- Solve from P_0 forward
- Online – Offline algorithm

FORWARD

- Solve problem at h with converged distribution
- Solve from P_0 forward
- Online – Offline algorithm



BACKWARD

- Solve problem at h with converged distribution
- Solve from P_h backward
- Offline algorithm

BACKWARD

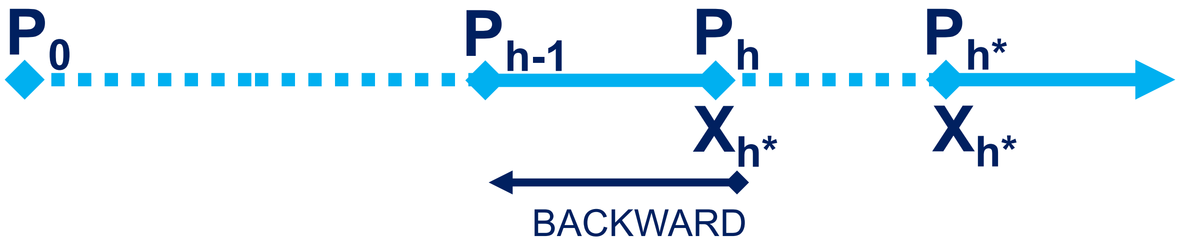
- Solve problem at h with converged distribution
- Solve from P_h backward
- Offline algorithm

BACKWARD

- Solve problem at h with converged distribution
- Solve from P_h backward
- Offline algorithm

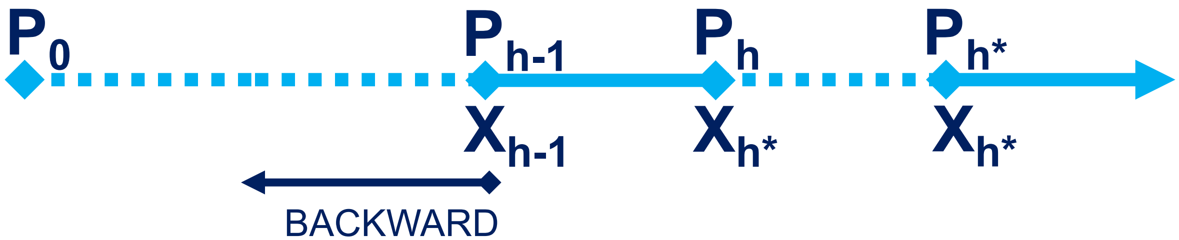
BACKWARD

- Solve problem at h with converged distribution
- Solve from P_h backward
- Offline algorithm



BACKWARD

- Solve problem at h with converged distribution
- Solve from P_h backward
- Offline algorithm



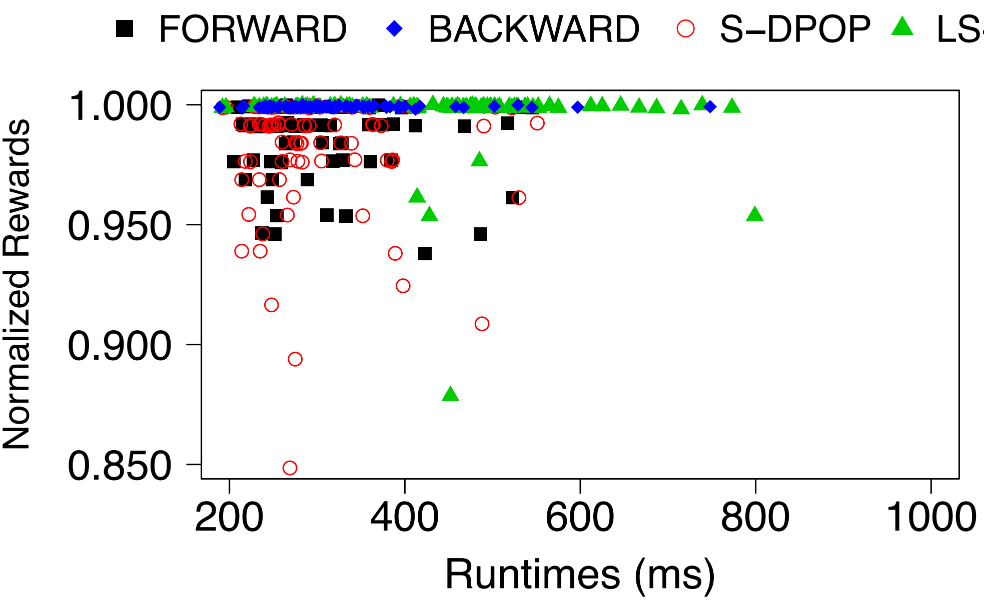
Content

- Distributed Constraint Optimization Problem
- Proactive Dynamic DCOPs
- Infinite-Horizon Proactive Dynamic DCOP
- Algorithms
- ➢ Experiments
- Conclusions

Experimental setup

- Random network
- Number of variables
  - Decision = Random: 8
- Horizon: 5
- Constraint density: 0.5
- Real distributed system, actual runtime

Experiment with Offline Algorithms



Switching cost = 100,000

S-DPOP:

- Not consider switching cost

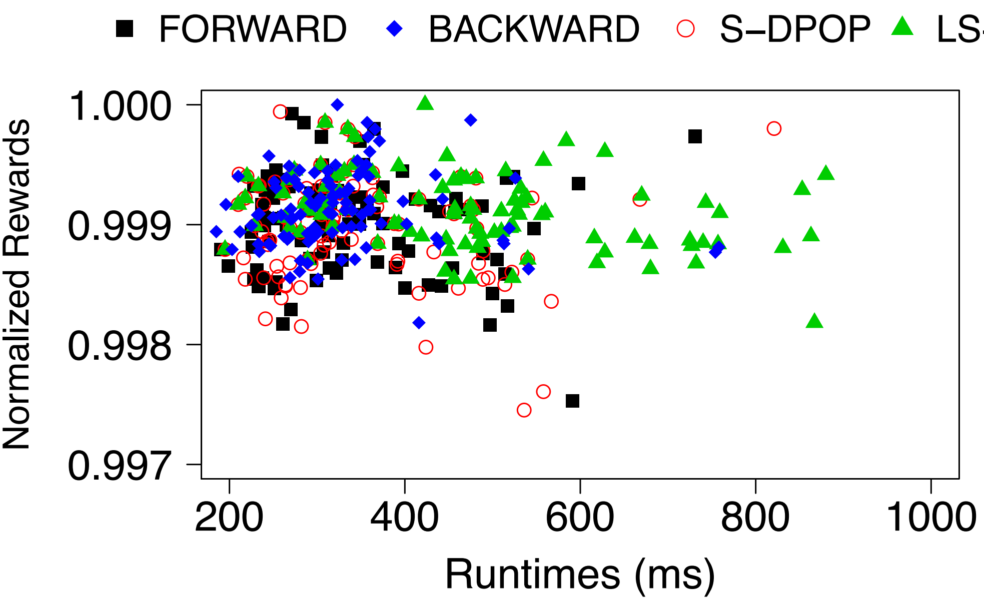
LS-SDPOP:

- Suffer from bad initial solutions

BACKWARD, FORWARD:

- Similar runtimes to S-DPOP
- Better solutions

Experiment with Offline Algorithms



Switching cost = 1,000

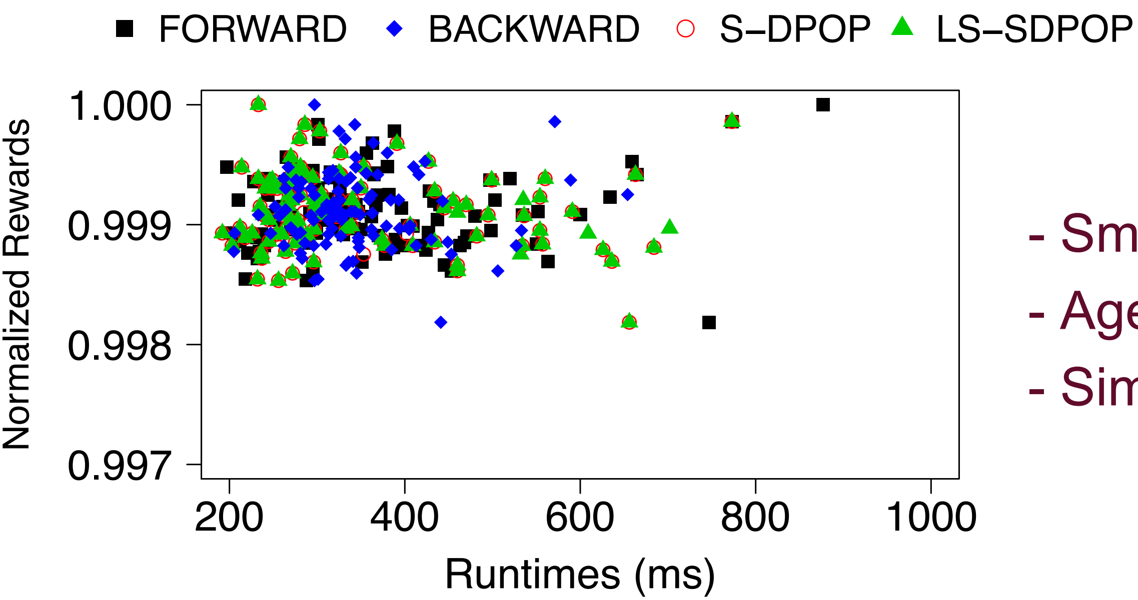
LS-SDPOP:

- Slower, better solutions

Other algorithms:

- Start to differ

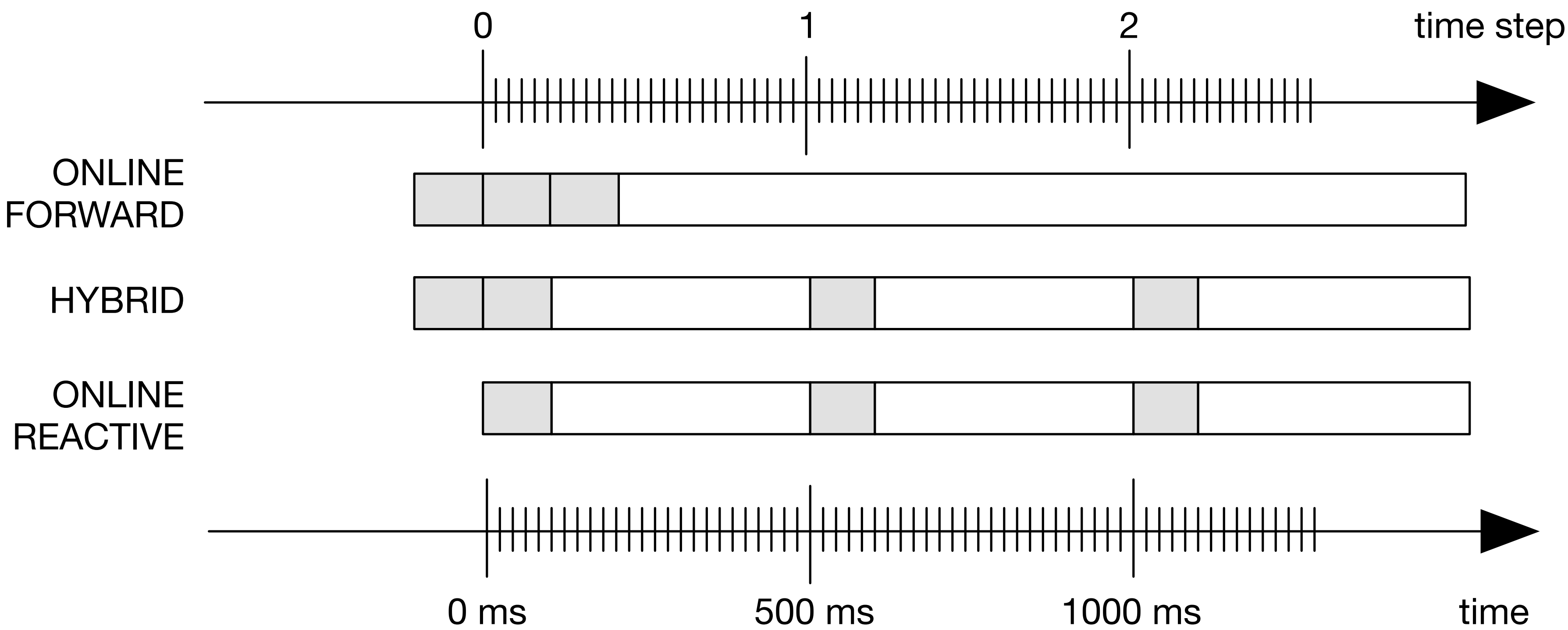
Experiment with Offline Algorithms



Switching cost = 10

- Small switching cost
- Agent sticks to initial solutions
- Similar quality, runtime

Experiment with Online Algorithms



Experiment with Online Algorithms

- Reactive

Experiment with Online Algorithms

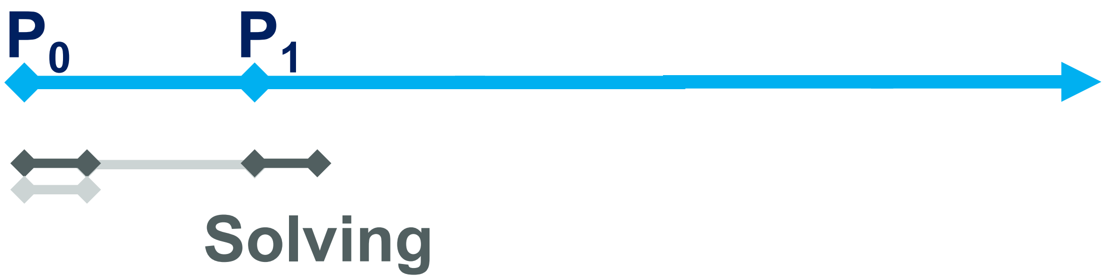
- Reactive

Experiment with Online Algorithms

- Reactive

Experiment with Online Algorithms

- Reactive

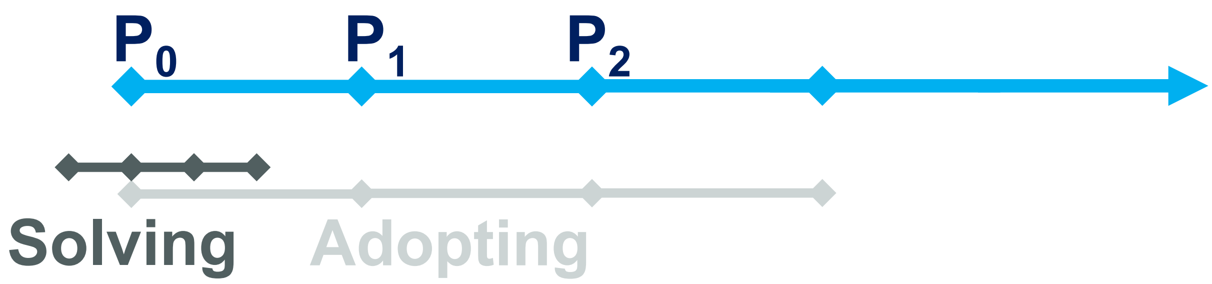


Experiment with Online Algorithms

- Reactive

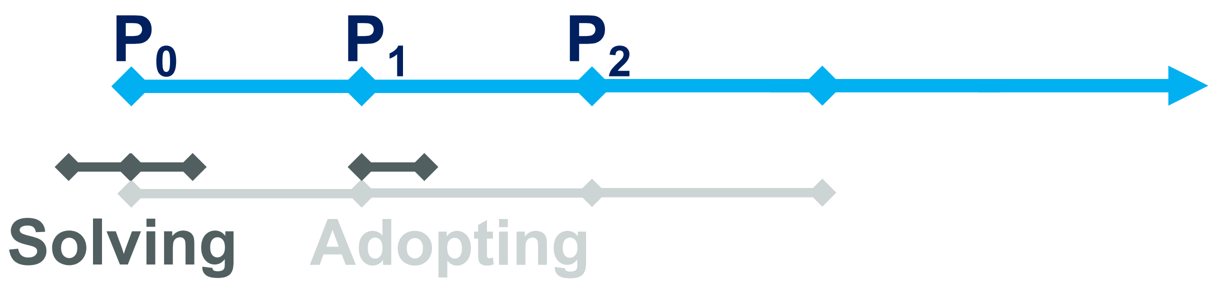
Experiment with Online Algorithms

- Online FORWARD

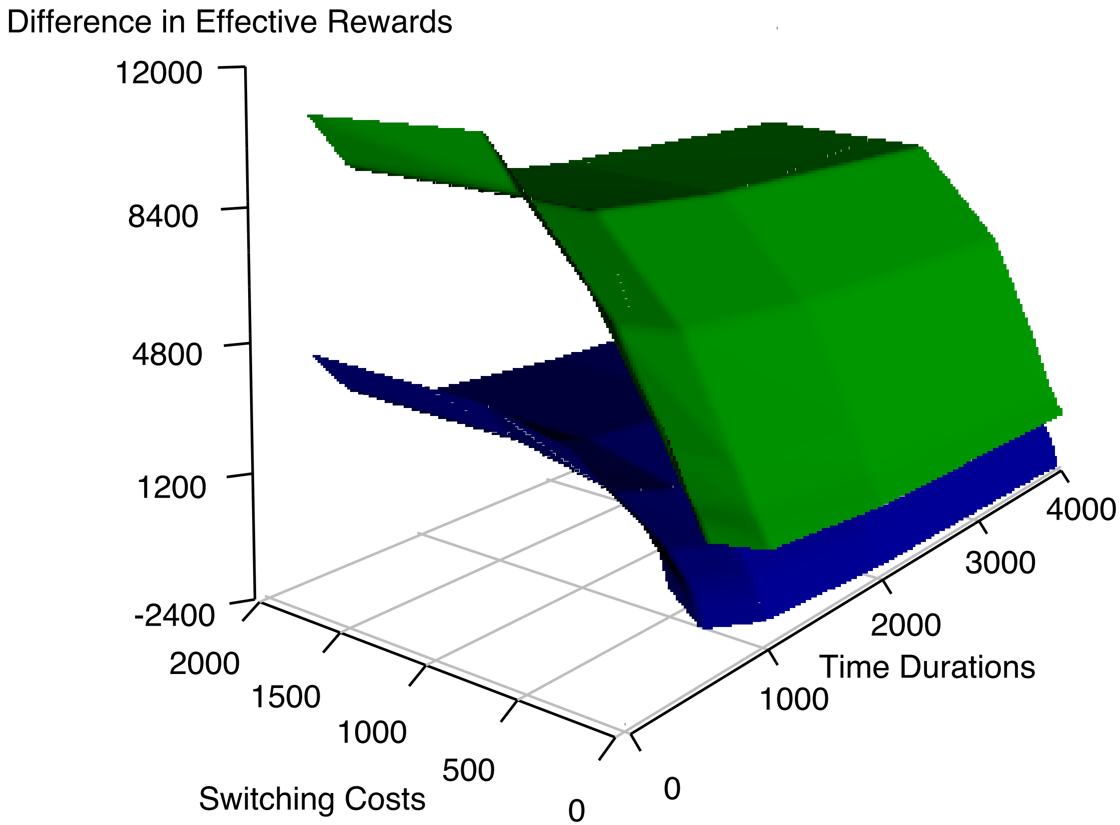


Experiment with Online Algorithms

- Online HYBRID



Experiment with Online Algorithms



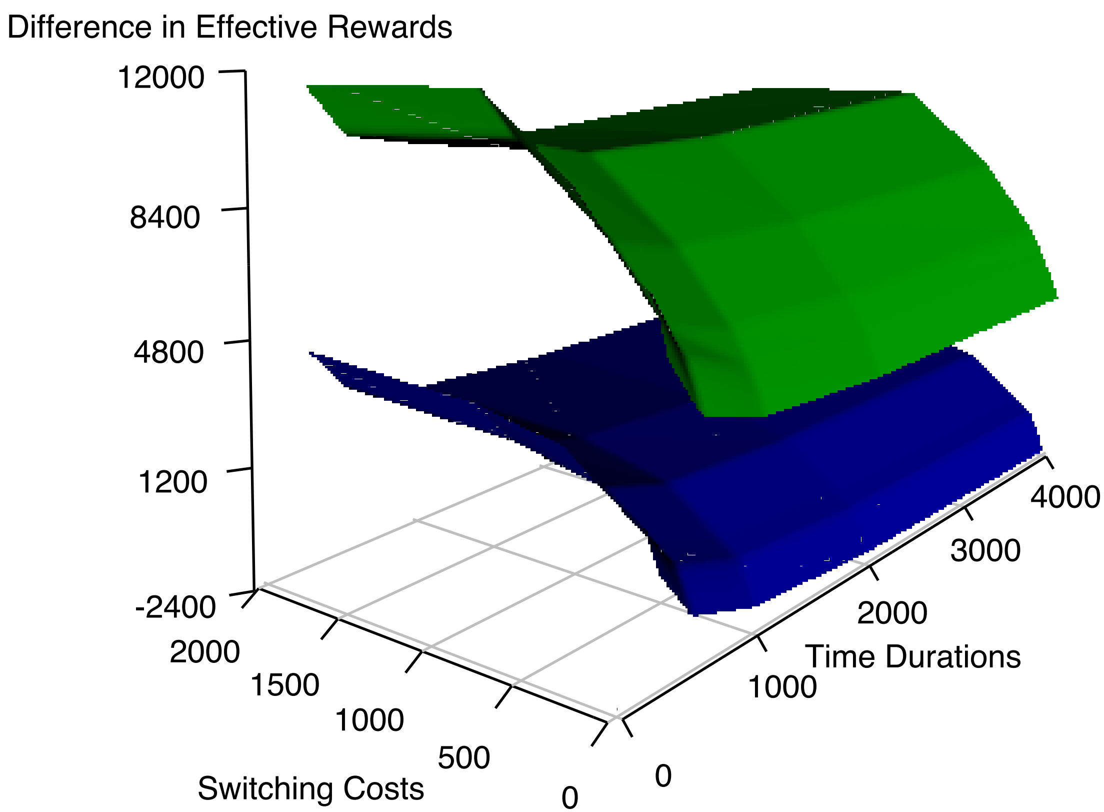
Reactive:

- Small switching cost
- Large time duration

FORWARD:

- Large switching cost
- Small time duration

Experiment with Online Algorithms



Similar results

HYBRID > FORWARD

Conclusions

- Infinite-Horizon Proactive Dynamic DCOP:
  - Optimal solution at time step h onwards
  - Random variables as Markov chains
  - Markov chain convergence
- Experiments:
  - Comparison between proactive and reactive dynamic DCOP algorithms (first time!)

Thank you

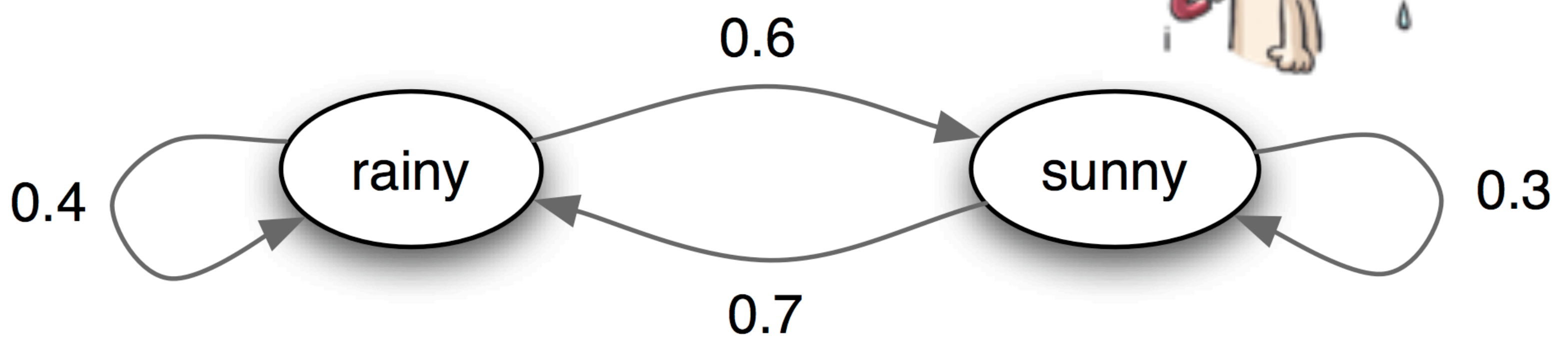
NM STATE UNIVERSITY

九州大学
KYUSHU UNIVERSITY

אוניברסיטת בן-גוריון בנגב
Ben-Gurion University of the Negev

UNIVERSITY OF MICHIGAN

Markov chains



- Markov property: Memoryless

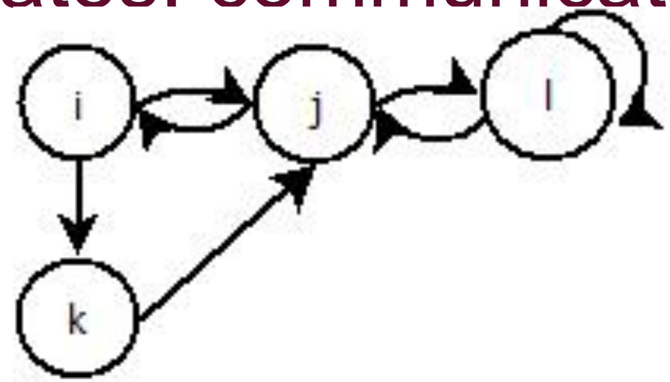
$$\Pr(x^t = j \mid x^{t-1} = i, x^{t-2} = r, \ldots, x^0 = s)$$
$$= \Pr(x^t = j \mid x^{t-1} = i)$$

Markov chain properties

- A state j is said **accessible** from state i (i -> j)

$$\Pr(x^t = j \mid x^0 = i) > 0$$

- State i and state j **communicate** (i <-> j)
- A **class** of states: communicate each other

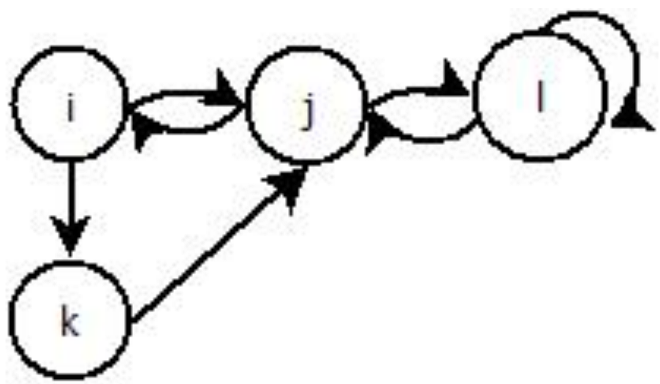


Markov chain properties

- **Period** of a state i:

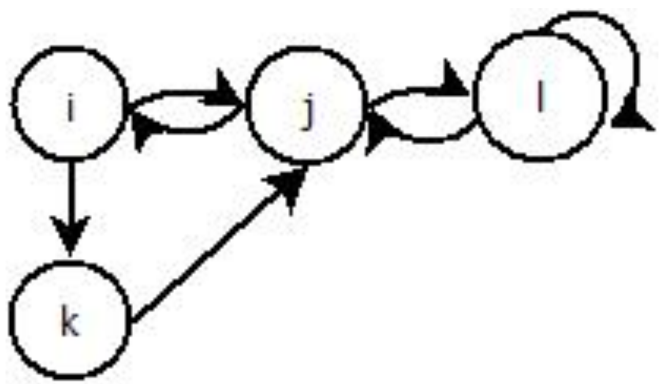
$$d(i) = \gcd\{t : P_{ii}^t > 0\}$$

- **Aperiodic**: period = 1



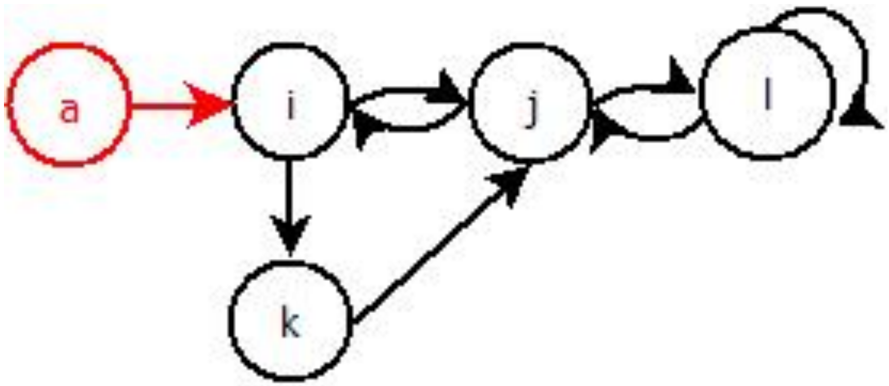
Markov chain properties

- **Recurrent**: (i -> j) => (j -> i)
- **Transient**: otherwise
- Ergodic: Both **aperiodic** and **recurrent**



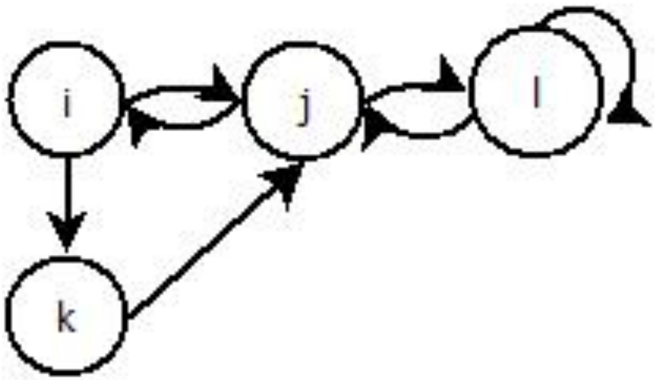
Markov chain properties

- **Recurrent**: (i -> j) => (j -> i)
- **Transient**: otherwise
- **Ergodic**: Both **recurrent** and **aperiodic**



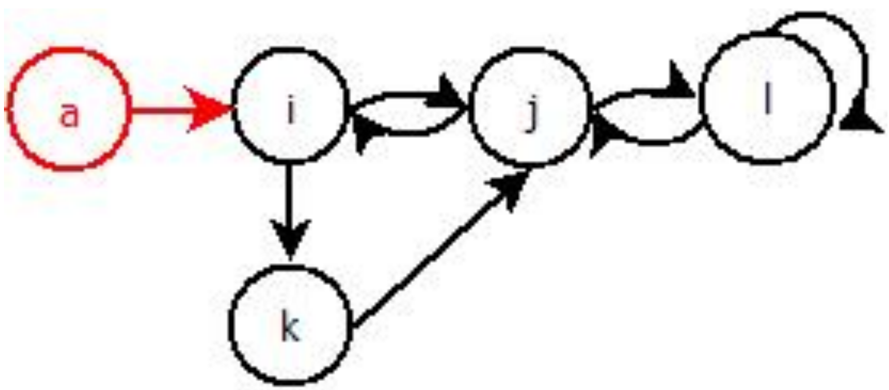
Markov chain properties

- **Unichain**: A chain that contains
  - Single recurrent class
  - Probably some transient states



Markov chain properties

- **Unichain**: A chain that contains
  - Single recurrent class
  - **Ergodic unichain**: aperiodic
  - Probably some transient states



Markov chains

- Convergence:

$$\pi^{t-1} \cdot P = \pi^{t} = \pi^{*}$$

- Conditions on convergence:

1. Positive transition matrix
2. All states: one single class and ergodic
3. The chain is an ergodic unichain

FORWARD

- Solve the last time step with stationary distribution

$$F^h(\mathbf{x}) = \sum_{\omega \in \Omega_y} f(\mathbf{x}|_{y=\omega}) \cdot p_y^*(\omega)$$

- Solve from time step 0

$$C^t(x) = -c \cdot \Delta(x^{t-1}, x^t)$$

- At time step h-1: $C^{h-1}(x) = -c \cdot \left(\Delta(x^{h-2}, x^{h-1}) + \Delta(x^{h-1}, x^*)\right)$
- Either online or offline algorithms

BACKWARD

- Solve the last time step with stationary distribution

$$F^h(\mathbf{x}) = \sum_{\omega \in \Omega_y} f(\mathbf{x}|_{y=\omega}) \cdot p_y^*(\omega)$$

- Solve from time step h-1 backwards

$$C^t(x) = -c \cdot \Delta(x^{t+1}, x^t)$$

- Offline algorithms