Minimizing the makespan of a project with stochastic activity durations under resource constraints

Stefan Creemers

(April 2, 2014)

IÉSEG SCHOOL OF MANAGEMENT

KATHOLIEKE UNIVERSITEIT LEUVEN

Agenda

- Problem setting:
  - Past work
  - Phase Type (PH) distributions
  - The SRCPSP
- Model discussion & comparison
- Results:
  - Solution quality
  - Computational performance
- Contribution

Problem setting

Creemers, Leus, Lambrecht (2010). Scheduling Markovian PERT networks to maximize the net present value, Operations Research Letters, 38, pp. 51-56.

Problem setting

Creemers, Leus, Lambrecht (2010). Scheduling Markovian PERT networks to maximize the net present value, Operations Research Letters, 38, pp. 51-56.

1. Maximum-eNPV objective
2. No resources
3. Exponentially-distributed activity durations
4. Use of a SDP recursion to obtain the optimal policy

Problem setting

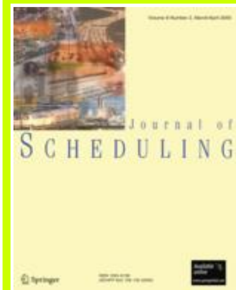
Creemers, Leus, Lambrecht (2010). Scheduling Markovian PERT networks to maximize the net present value, Operations Research Letters, 38, pp. 51-56.

Creemers (under review) Minimizing the makespan of a project with stochastic activity durations under resource constraints, Journal of Scheduling.

1. Maximum-eNPV objective
2. No resources
3. Exponentially-distributed activity durations
4. Use of a SDP recursion to obtain the optimal policy

Problem setting

Creemers, Leus, Lambrecht (2010). Scheduling Markovian PERT networks to maximize the net present value, Operations Research Letters, 38, pp. 51-56.



Creemers (under review) Minimizing the makespan of a project with stochastic activity durations under resource constraints, Journal of Scheduling.

1. Maximum-eNPV objective
2. No resources
3. Exponentially-distributed activity durations
4. Use of a SDP recursion to obtain the optimal policy

1. Minimum-makespan objective
2. Renewable resources
3. General activity durations (PH approximation)
4. Use of an improved/modified SDP recursion

Problem setting

Creemers, Leus, Lambrecht (2010). Scheduling Markovian PERT networks to maximize the net present value, Operations Research Letters, 38, pp. 51-56.

Creemers (under review) Minimizing the makespan of a project with stochastic activity durations under resource constraints, Journal of Scheduling.

1. Maximum-eNPV objective
2. No resources
3. Exponentially-distributed activity durations
4. Use of a SDP recursion to obtain the optimal policy

1. Minimum-makespan objective
2. Renewable resources
3. General activity durations (PH approximation)
4. **Use of an improved/modified SDP recursion**

Improvement of the SDP recursion

n	OS	% Solved	Average CPU (2010)	Average CPU (improved)	Average Factor
10	0.8	100%	0.00	0.00	**-**
10	0.6	100%	0.00	0.00	**-**
10	0.4	100%	0.00	0.00	**6.81**
20	0.8	100%	0.00	0.00	**-**
20	0.6	100%	0.01	0.00	**27.25**
20	0.4	100%	0.46	0.03	**17.60**
30	0.8	100%	0.01	0.00	**17.53**
30	0.6	100%	0.33	0.02	**14.90**
30	0.4	100%	26.92	1.49	**18.12**
40	0.8	100%	0.03	0.00	**12.41**
40	0.6	100%	6.62	0.49	**13.62**
40	0.4	97%	2,337.96	72.25	**32.36**
50	0.8	100%	0.15	0.01	**10.60**
50	0.6	100%	100.28	4.43	**22.62**
50	0.4	13%	52,267.30	823.71	**63.45**
60	0.8	100%	0.74	0.06	**12.36**
60	0.6	100%	2,210.08	67.87	**32.56**
60	0.4	0%	-	-	**-**

n	OS	% Solved	Average CPU (2010)	Average CPU (improved)	Average Factor
70	0.8	100%	3.19	0.24	**13.09**
70	0.6	73%	17,495.49	378.64	**46.21**
70	0.4	0%	-	-	**-**
80	0.8	100%	10.81	0.79	**13.65**
80	0.6	30%	72,473.41	1,188.01	**61.00**
80	0.4	0%	-	-	**-**
90	0.8	100%	50.64	3.15	**16.06**
90	0.6	0%	-	-	**-**
90	0.4	0%	-	-	**-**
100	0.8	100%	171.42	9.60	**17.85**
100	0.6	0%	-	-	**-**
100	0.4	0%	-	-	**-**
110	0.8	100%	1,193.88	40.93	**29.17**
110	0.6	0%	-	-	**-**
110	0.4	0%	-	-	**-**
120	0.8	100%	12,789.06	260.66	**49.06**
120	0.6	0%	-	-	**-**
120	0.4	0%	-	-	**-**

Improvement of the SDP recursion

n	OS	% Solved	Average CPU (2010)	Average CPU (improved)	Average Factor
10	0.8	100%	0.00	0.00	-
10	0.6	100%	0.00	0.00	-
10	0.4	100%	0.00	0.00	**6.81**
20	0.8	[illegible]	[illegible]	[illegible]	[illegible]
20	0.6	[illegible]	[illegible]	[illegible]	[illegible]
20	0.4	[illegible]	[illegible]	[illegible]	[illegible]
30	0.8	[illegible]	[illegible]	[illegible]	[illegible]
30	0.6	[illegible]	[illegible]	[illegible]	[illegible]
30	0.4	[illegible]	[illegible]	[illegible]	[illegible]
40	0.8	[illegible]	[illegible]	[illegible]	[illegible]
40	0.6	[illegible]	[illegible]	[illegible]	[illegible]
40	0.4	[illegible]	[illegible]	[illegible]	[illegible]
50	0.8	[illegible]	[illegible]	[illegible]	[illegible]
50	0.6	100%	100.28	4.43	**22.62**
50	0.4	13%	52,267.30	823.71	**63.45**
60	0.8	100%	0.74	0.06	**12.36**
60	0.6	100%	2,210.08	67.87	**32.56**
60	0.4	0%	-	-	-

n	OS	% Solved	Average CPU (2010)	Average CPU (improved)	Average Factor
70	0.8	100%	3.19	0.24	**13.09**
70	0.6	73%	17,495.49	378.64	**46.21**
70	0.4	0%	-	-	-
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	**13.65**
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	**61.00**
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	**16.06**
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	**17.85**
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	**29.17**
110	0.6	0%	-	-	-
110	0.4	0%	-	-	-
120	0.8	100%	12,789.06	260.66	**49.06**
120	0.6	0%	-	-	-
120	0.4	0%	-	-	-

In comparison with the model of Creemers et al. (2010), the computation speed has been increased by factor 50 (= **5,000% faster**).

Improvement of the SDP recursion

n	OS	% Solved	Average CPU (2010)	Average CPU (improved)	Average Factor
10	0.8	100%	0.00	0.00	-
10	0.6	100%	0.00	0.00	-
10	0.4	100%	0.00	0.00	6.81
20	0.8				
20	0.6				
20	0.4				
30	0.8				
30	0.6				
30	0.4				
40	0.8				
40	0.6				
40	0.4				
50	0.8				
50	0.6	100%	100.28	4.43	22.62
50	0.4	13%	52,267.30	823.71	63.45
60	0.8	100%	0.74	0.06	12.36
60	0.6	100%	2,210.08	67.87	32.56
60	0.4	0%	-	-	-

n	OS	% Solved	Average CPU (2010)	Average CPU (improved)	Average Factor
70	0.8	100%	3.19	0.24	13.09
70	0.6	73%	17,495.49	378.64	46.21
70	0.4	0%	-	-	-
					13.65
					61.00
					-
					16.06
					-
					-
					17.85
					-
					-
					29.17
110	0.6	0%	-	-	-
110	0.4	0%	-	-	-
120	0.8	100%	12,789.06	260.66	49.06
120	0.6	0%	-	-	-
120	0.4	0%	-	-	-

In comparison with the model of Creemers et al. (2010), the computation speed has been increased by factor 50 (= **5,000% faster**).

When compared to the model of Sobel et al. (2009), the new model is even **750,000% faster**.

Problem setting

Creemers, Leus, Lambrecht (2010). Scheduling Markovian PERT networks to maximize the net present value, Operations Research Letters, 38, pp. 51-56.

1. Maximum-eNPV objective
2. No resources
3. Exponentially-distributed activity durations
4. Use of a SDP recursion to obtain the optimal policy

Creemers (under review) Minimizing the makespan of a project with stochastic activity durations under resource constraints, Journal of Scheduling.

1. Minimum-makespan objective
2. Renewable resources
3. **General activity durations (PH approximation)**
4. Use of an improved/modified SDP recursion

Model extensions: PH distributions

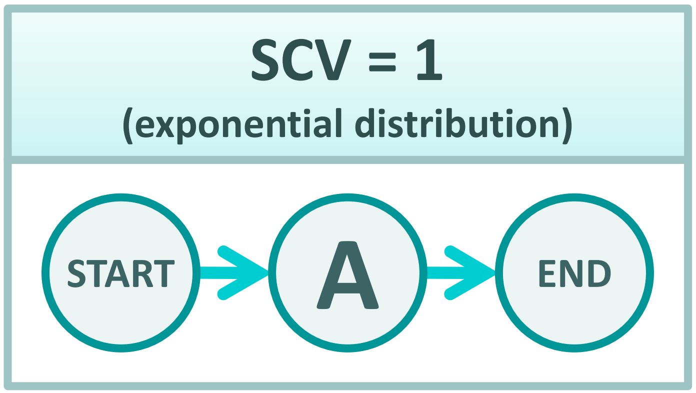
- Introduced by Neuts in 1981
- A Phase Type (PH) distribution is a mixture of exponential distributions
- The exponential, Erlang, Coxian, and hyper-exponential distribution are all examples of a PH distribution
- We use simple PH distributions to match the first two moments of the distribution of the activity duration (more advanced PH distributions, however, can also be used)

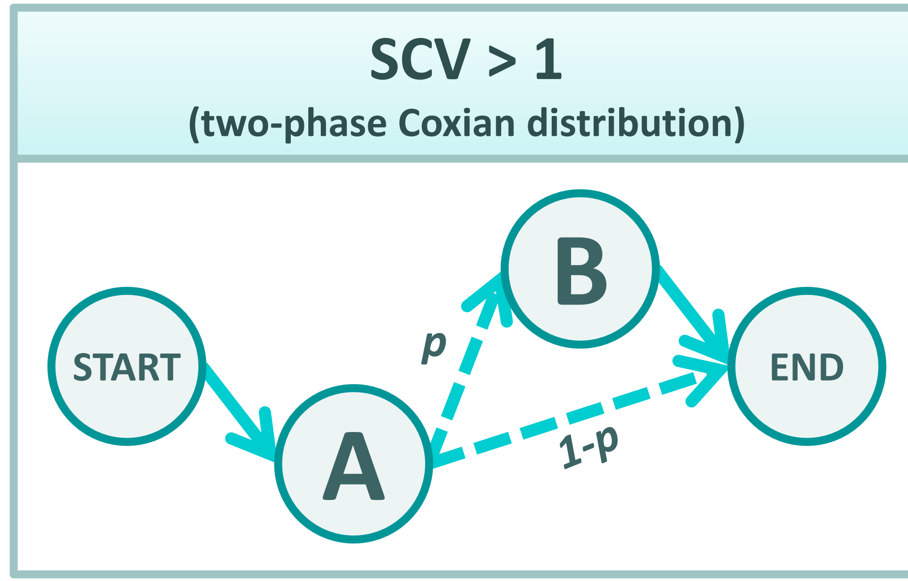
PH distributions: Example of a single activity

PH distributions: Example of a single activity

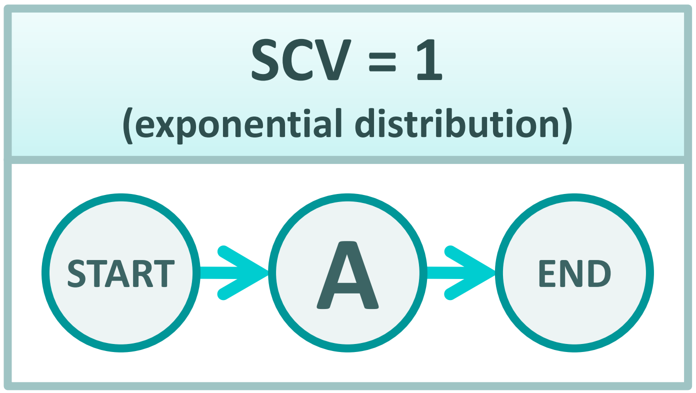
SCV = 1
(exponential distribution)

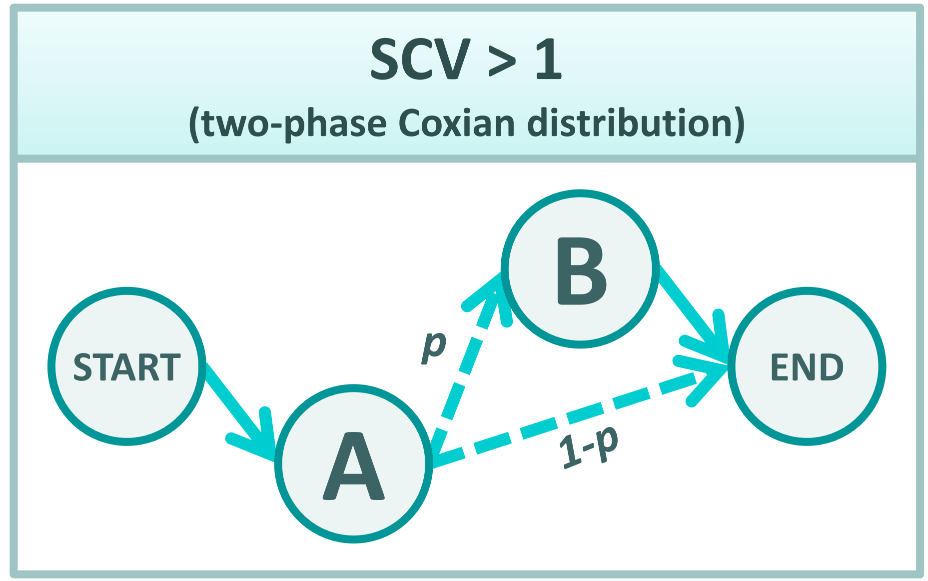
PH distributions: Example of a single activity

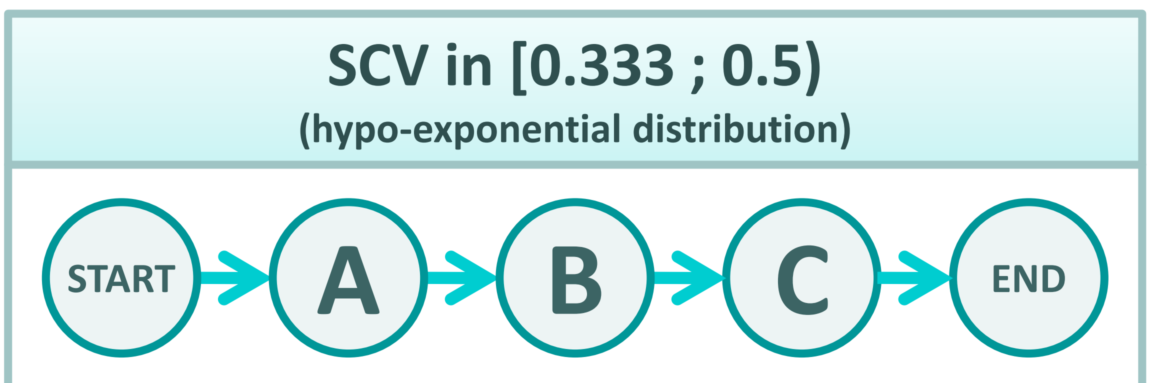




PH distributions: Example of a single activity

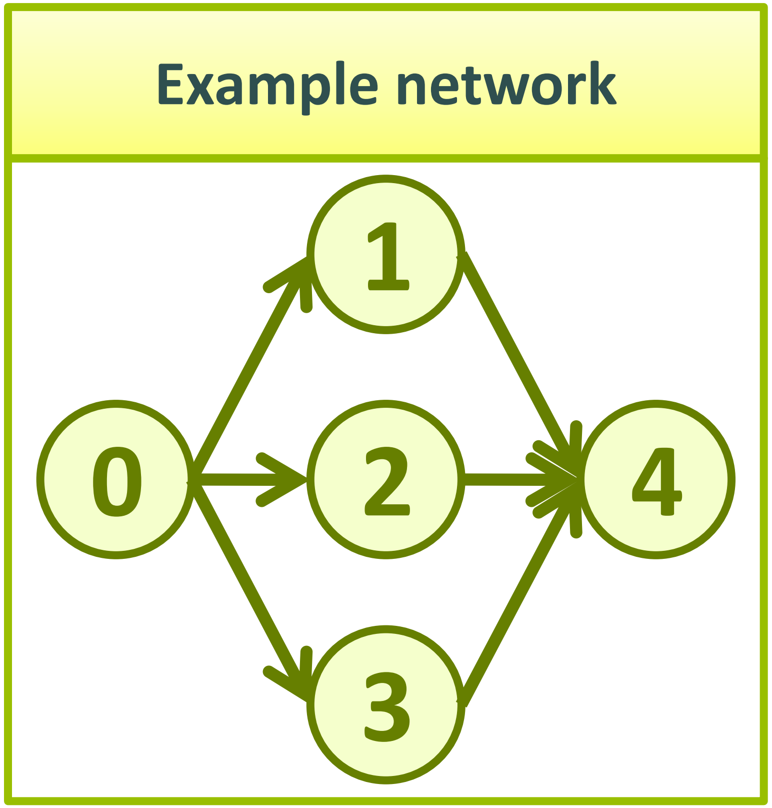






PH distributions: Example of a project network

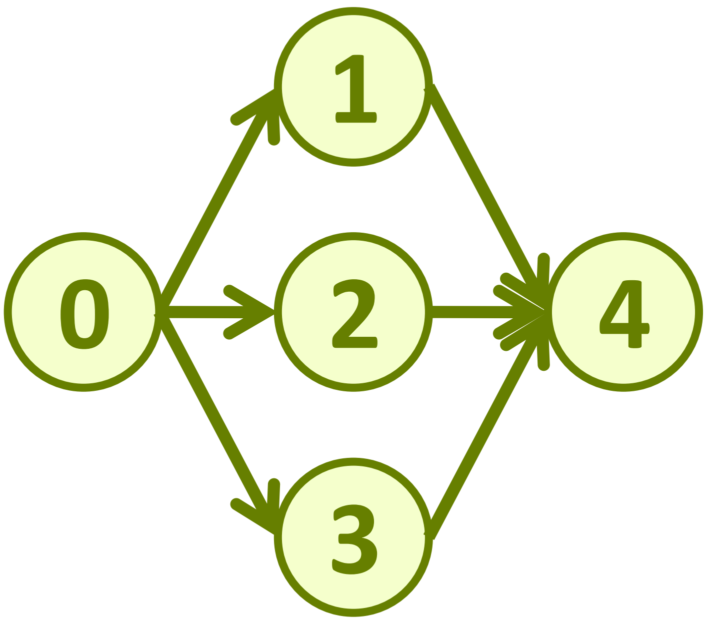
PH distributions: Example of a project network



PH distributions: Example of a project network

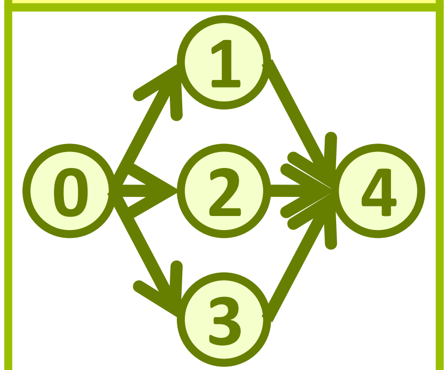
Activity	SCV
0	Dummy start
1	SCV in [0.33;0.5)
2	SCV = 1
3	SCV > 1
4	Dummy finish

Example network



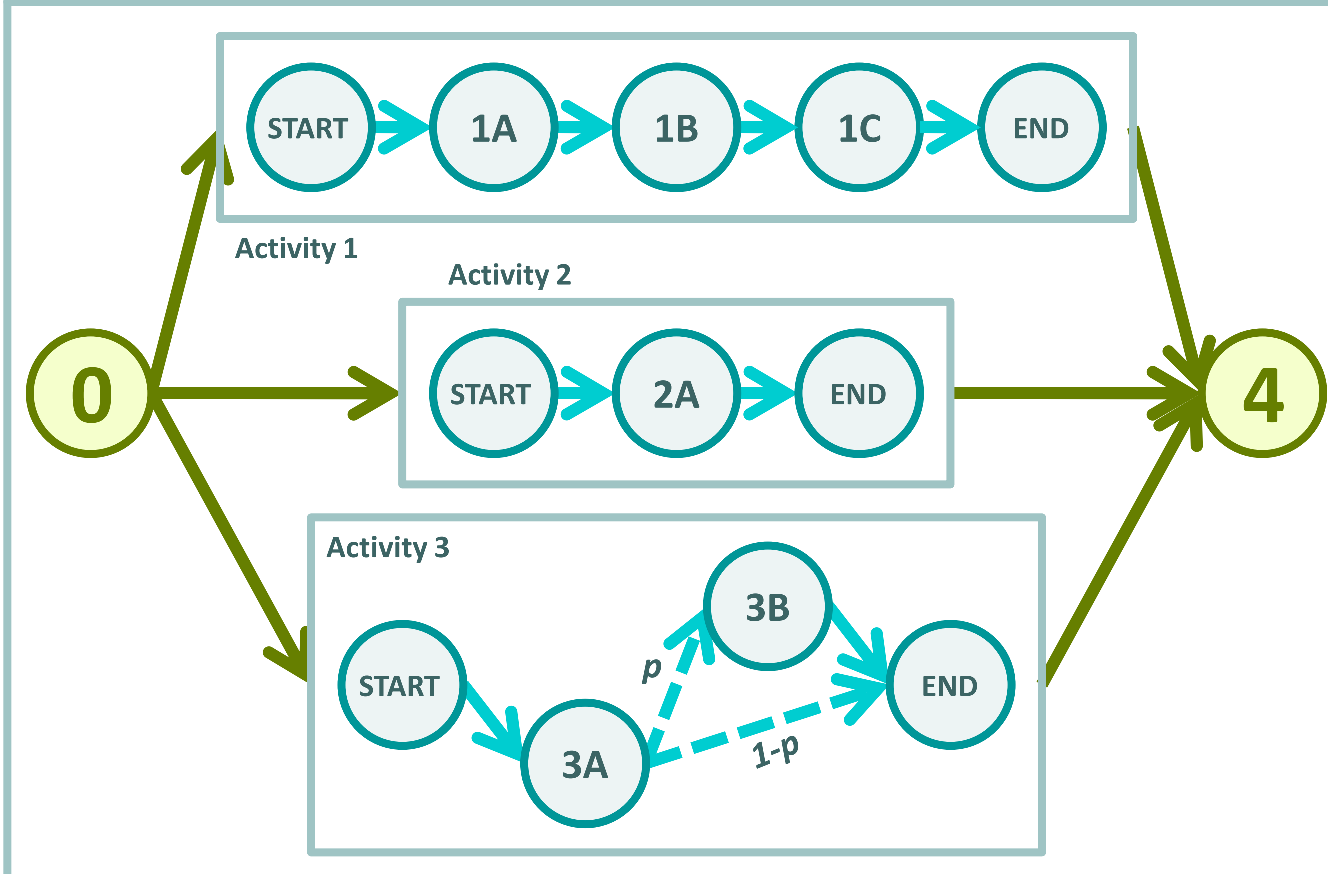
PH distributions: Example of a project network

Example network

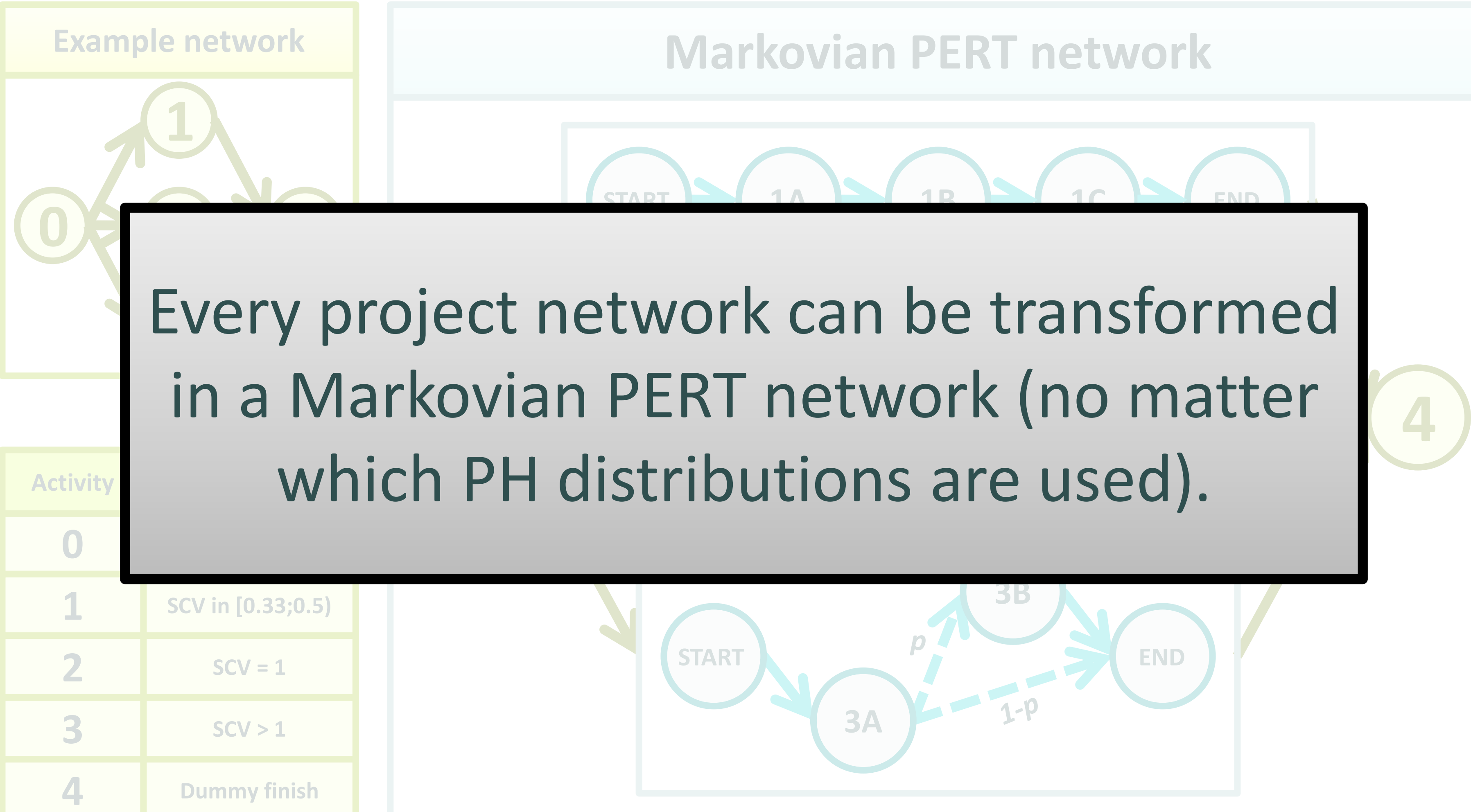


Activity	SCV
0	Dummy start
1	SCV in [0.33;0.5)
2	SCV = 1
3	SCV > 1
4	Dummy finish

Markovian PERT network



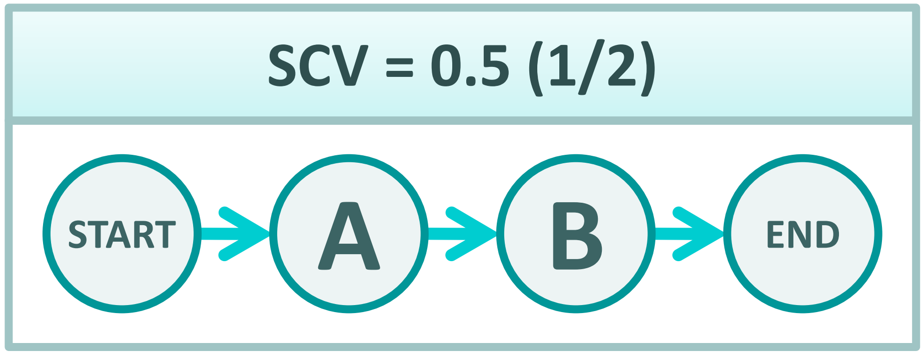
PH distributions: Example of a project network



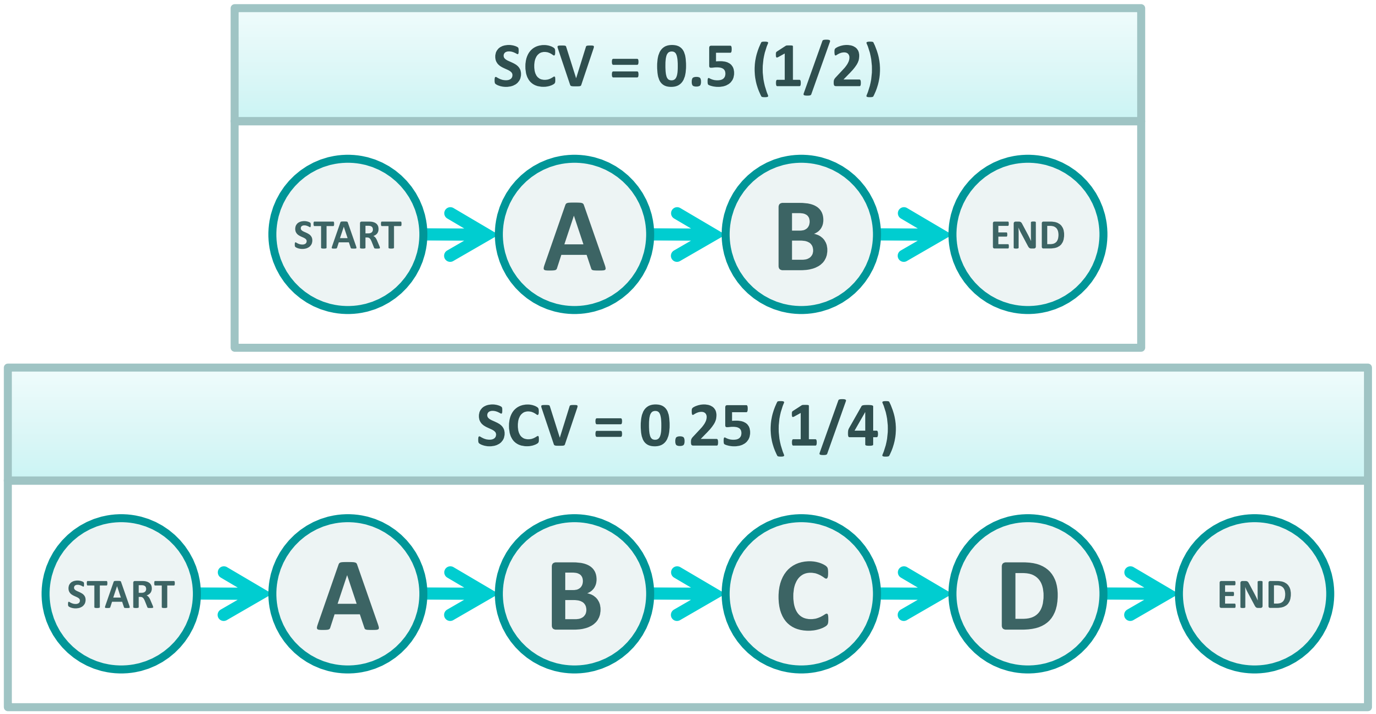
Every project network can be transformed in a Markovian PERT network (no matter which PH distributions are used).

PH distributions: What about low variability?

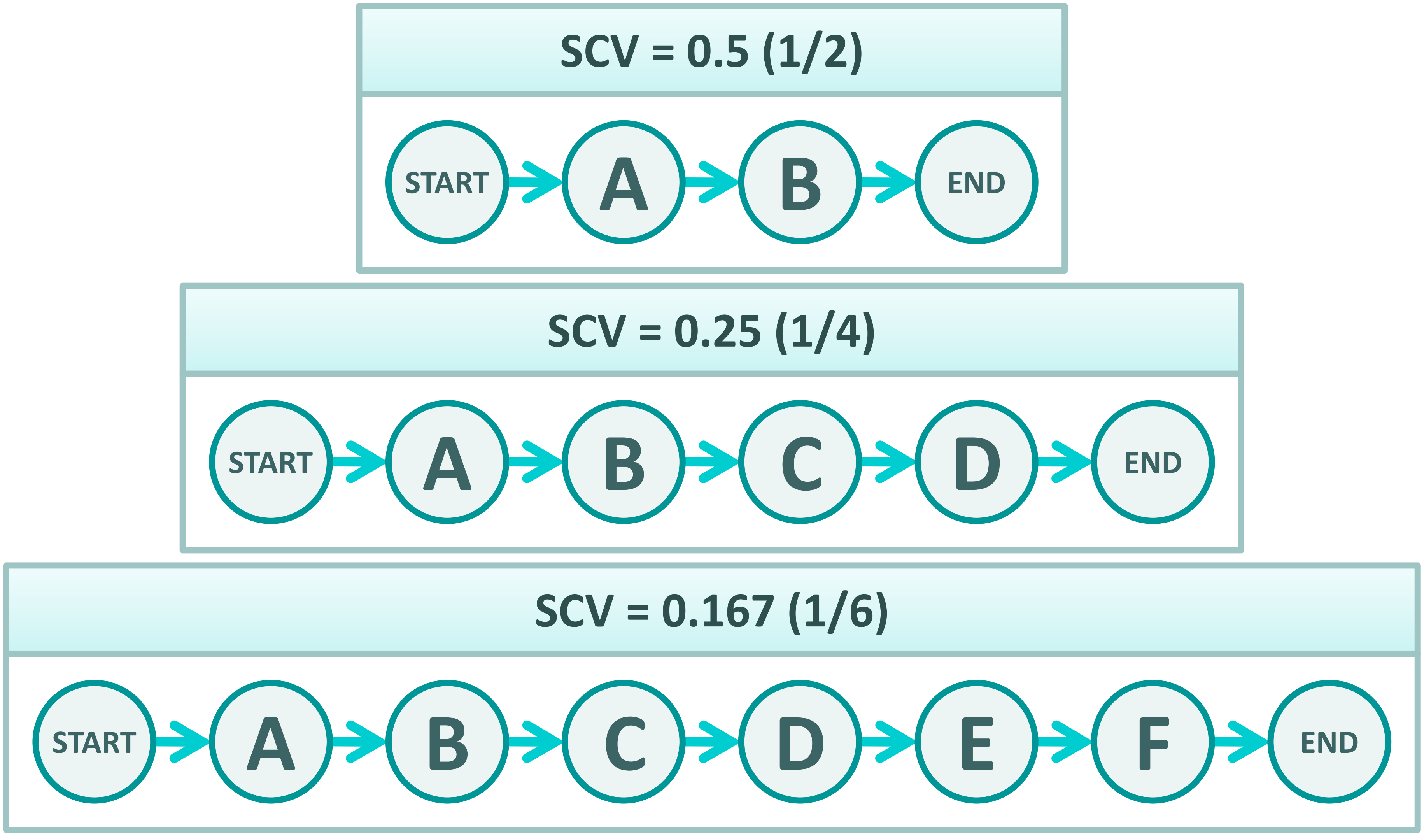
PH distributions: What about low variability?



PH distributions: What about low variability?



PH distributions: What about low variability?



PH distributions: What about low variability?

Low variability duration variability inflates the size of the Markovian PERT network.

=>

Our model works best when duration variability is moderate to high.

Problem setting

Creemers, Leus, Lambrecht (2010). Scheduling Markovian PERT networks to maximize the net present value, Operations Research Letters, 38, pp. 51-56.

1. Maximum-eNPV objective
2. No resources
3. Exponentially-distributed activity durations
4. Use of a SDP recursion to obtain the optimal policy

Creemers (under review) Minimizing the makespan of a project with stochastic activity durations under resource constraints, Journal of Scheduling.

1. **Minimum-makespan objective**
2. **Renewable resources**
3. General activity durations (PH approximation)
4. Use of an improved/modified SDP recursion

Literature on the SRCPSP

Literature on the SRCPSP

Author	Tsai and Gemmill (1998)
Date	1998
Method	Simulated annealing & tabu search
Policy class	RB (Resource-Based)

Literature on the SRCPSP

Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)
Date	1998	2009
Method	Simulated annealing & tabu search	Genetic algorithm
Policy class	RB (Resource-Based)	AB (Activity-Based)

Literature on the SRCPSP

Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)	Ashtiani et al. (2011)
Date	1998	2009	2011
Method	Simulated annealing & tabu search	Genetic algorithm	Two-phase local-search procedure
Policy class	RB (Resource-Based)	AB (Activity-Based)	PP (PreProcessor)

Literature on the SRCPSP

	Heuristic approaches		
Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)	Ashtiani et al. (2011)
Date	1998	2009	2011
Method	Simulated annealing & tabu search	Genetic algorithm	Two-phase local-search procedure
Policy class	RB (Resource-Based)	AB (Activity-Based)	PP (PreProcessor)

Literature on the SRCPSP

	Heuristic approaches			
Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)	Ashtiani et al. (2011)	Stork (2001)
Date	1998	2009	2011	2001
Method	Simulated annealing & tabu search	Genetic algorithm	Two-phase local-search procedure	Five B&B algorithms
Policy class	RB (Resource-Based)	AB (Activity-Based)	PP (PreProcessor)	AB & ES

Literature on the SRCPSP

	Heuristic approaches				
Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)	Ashtiani et al. (2011)	Stork (2001)	Creemers (201?)
Date	1998	2009	2011	2001	Under review
Method	Simulated annealing & tabu search	Genetic algorithm	Two-phase local-search procedure	Five B&B algorithms	SDP recursion
Policy class	RB (Resource-Based)	AB (Activity-Based)	PP (PreProcessor)	AB & ES	Elementary policies

Literature on the SRCPSP

	Heuristic approaches			Optimal approaches	
Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)	Ashtiani et al. (2011)	Stork (2001)	Creemers (201?)
Date	1998	2009	2011	2001	Under review
Method	Simulated annealing & tabu search	Genetic algorithm	Two-phase local-search procedure	Five B&B algorithms	SDP recursion
Policy class	RB (Resource-Based)	AB (Activity-Based)	PP (PreProcessor)	AB & ES	Elementary policies

Literature on the SRCPSP

	Heuristic approaches			Optimal approaches	
Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)	Ashtiani et al. (2011)	Stork (2001)	Creemers (201?)
Date	1998	2009	2011	2001	Under review
Method	Simulated annealing & tabu search	Genetic algorithm	Two-phase local-search procedure	Five B&B algorithms	SDP recursion
Policy class	RB (Resource-Based)	AB (Activity-Based)	PP (PreProcessor)	AB & ES	Elementary policies
Data set					

Literature on the SRCPSP

	Heuristic approaches			Optimal approaches	
Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)	Ashtiani et al. (2011)	Stork (2001)	Creemers (201?)
Date	1998	2009	2011	2001	Under review
Method	Simulated annealing & tabu search	Genetic algorithm	Two-phase local-search procedure	Five B&B algorithms	SDP recursion
Policy class	RB (Resource-Based)	AB (Activity-Based)	PP (PreProcessor)	AB & ES	Elementary policies
Data set					
J30 (PSPLIB)					
J60 (PSPLIB)					
J120 (PSPLIB)					
Patterson					
Golenko					

Literature on the SRCPSP

	Heuristic approaches			Optimal approaches	
Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)	Ashtiani et al. (2011)	Stork (2001)	Creemers (201?)
Date	1998	2009	2011	2001	Under review
Method	Simulated annealing & tabu search	Genetic algorithm	Two-phase local-search procedure	Five B&B algorithms	SDP recursion
Policy class	RB (Resource-Based)	AB (Activity-Based)	PP (PreProcessor)	AB & ES	Elementary policies
Data set					
J30 (PSPLIB)					
J60 (PSPLIB)					
J120 (PSPLIB)					
Patterson					
Golenko					

Literature on the SRCPSP

	Heuristic approaches			Optimal approaches	
Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)	Ashtiani et al. (2011)	Stork (2001)	Creemers (201?)
Date	1998	2009	2011	2001	Under review
Method	Simulated annealing & tabu search	Genetic algorithm	Two-phase local-search procedure	Five B&B algorithms	SDP recursion
Policy class	RB (Resource-Based)	AB (Activity-Based)	PP (PreProcessor)	AB & ES	Elementary policies
Data set					
J30 (PSPLIB)					
J60 (PSPLIB)					
J120 (PSPLIB)					
Patterson					
Golenko					

Literature on the SRCPSP

	Heuristic approaches			Optimal approaches	
Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)	Ashtiani et al. (2011)	Stork (2001)	Creemers (201?)
Date	1998	2009	2011	2001	Under review
Method	Simulated annealing & tabu search	Genetic algorithm	Two-phase local-search procedure	Five B&B algorithms	SDP recursion
Policy class	RB (Resource-Based)	AB (Activity-Based)	PP (PreProcessor)	AB & ES	Elementary policies
Data set					
J30 (PSPLIB)					
J60 (PSPLIB)					
J120 (PSPLIB)					**IMPOSSIBLE**
Patterson					
Golenko					

Literature on the SRCPSP

	Heuristic approaches			Optimal approaches	
Author	Tsai and Gemmill (1998)	Ballestìn & Leus (2009)	Ashtiani et al. (2011)	Stork (2001)	Creemers (201?)
Date	1998	2009	2011	2001	Under review
Method	Simulated annealing & tabu search	Genetic algorithm	Two-phase local-search procedure	Five B&B algorithms	SDP recursion
Policy class	RB (Resource-Based)	AB (Activity-Based)	PP (PreProcessor)	AB & ES	Elementary policies
Data set					
J30 (PSPLIB)					
J60 (PSPLIB)					
J120 (PSPLIB)					**IMPOSSIBLE**
Patterson					
Golenko					**SCV = 0.014**

Results: Solution quality

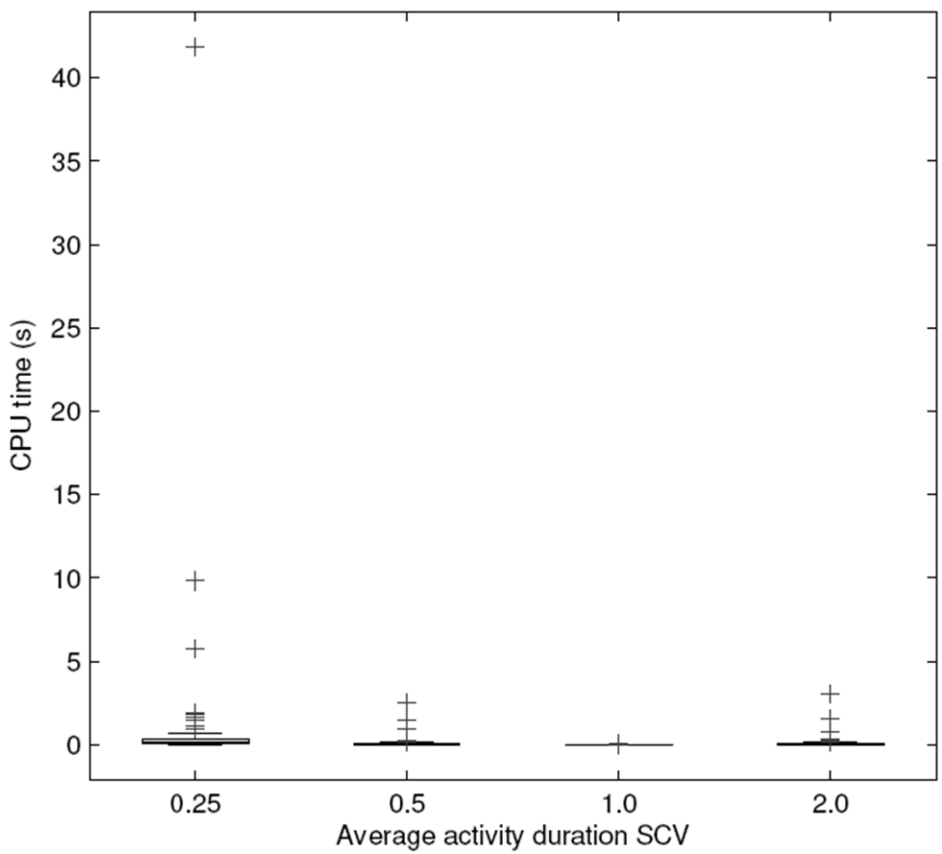
Results: Solution quality

- We optimize over a more general class of policies => we expect better results.
- From Ballestìn & Leus (2009) we obtained the results for the J30 & J60 problem instances if activity durations are exponentially distributed:
  - J30 average improvement of solution quality of 13,2%
  - J60 average improvement of solution quality of 13,5%

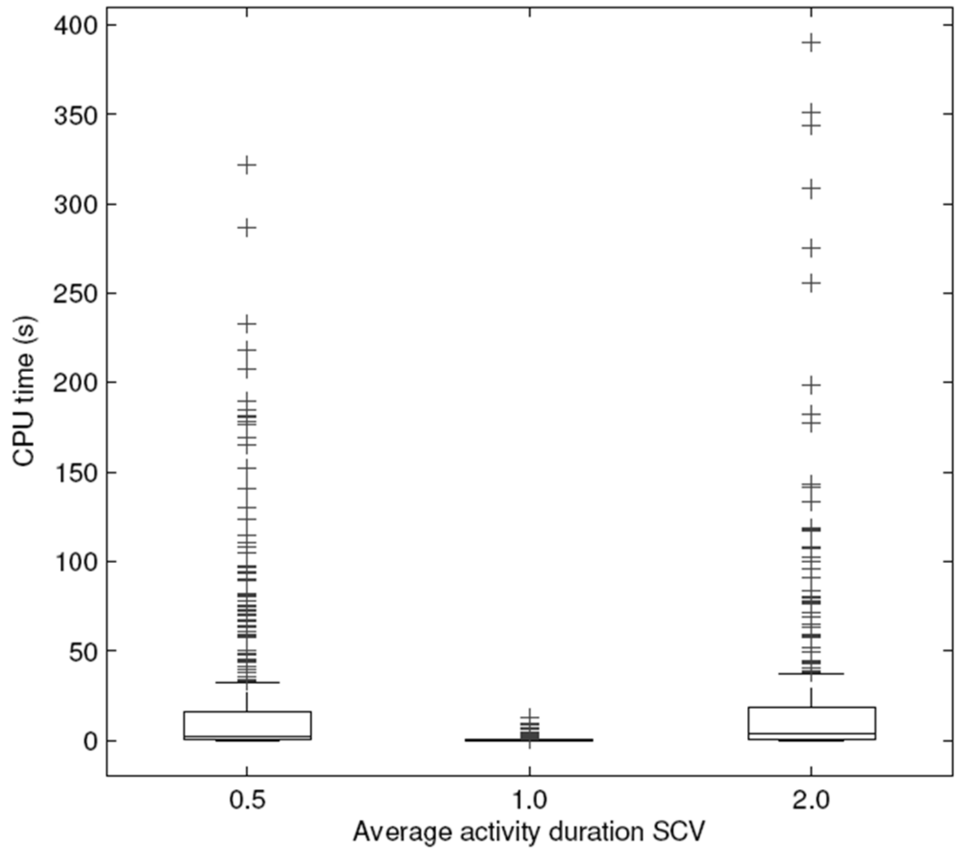
=> Significant improvement of solution quality!

Results: Computational performance

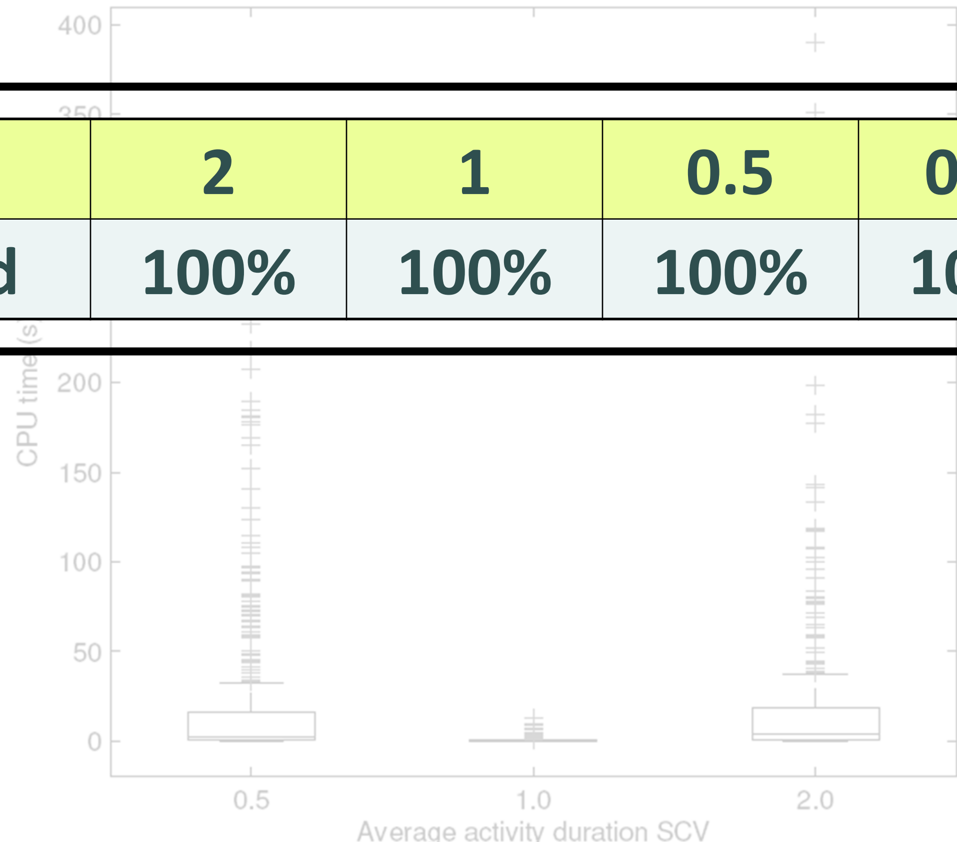
Results: Computational performance (Patterson)



Results: Computational performance (J30 - PSPLIB)

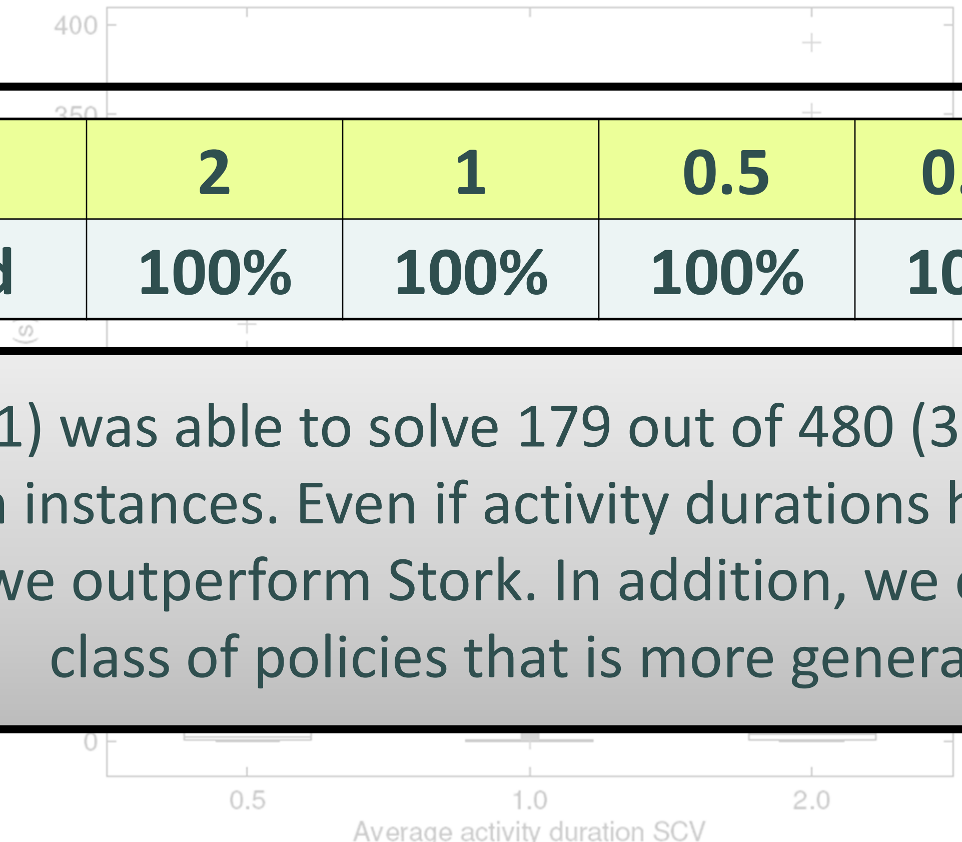


Results: Computational performance (J30 - PSPLIB)



SCV	2	1	0.5	0.33	0.25
% Solved	100%	100%	100%	100%	75%

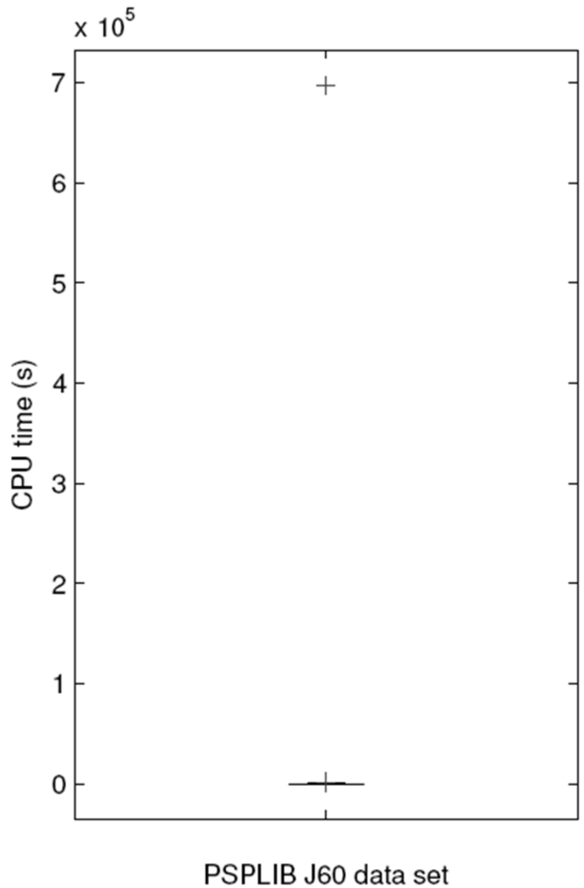
Results: Computational performance (J30 - PSPLIB)



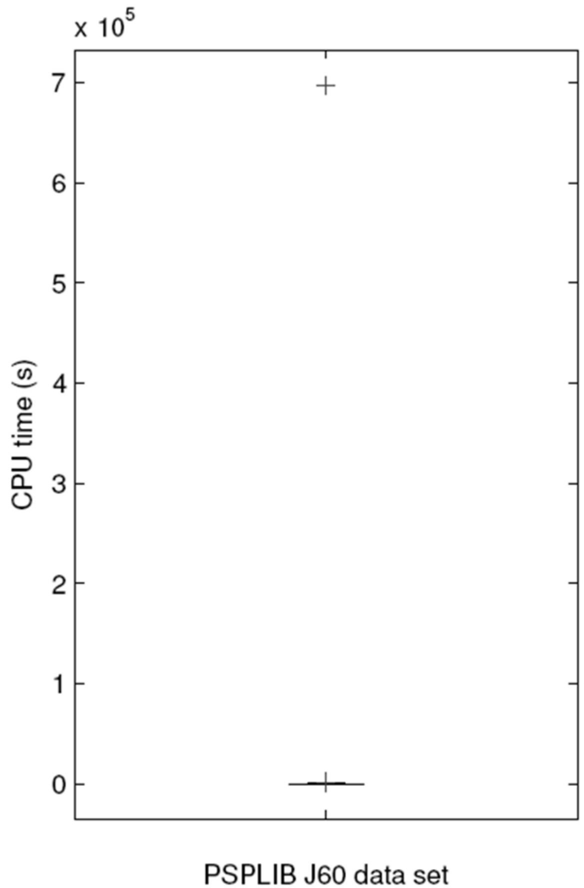
SCV	2	1	0.5	0.33	0.25
% Solved	100%	100%	100%	100%	75%

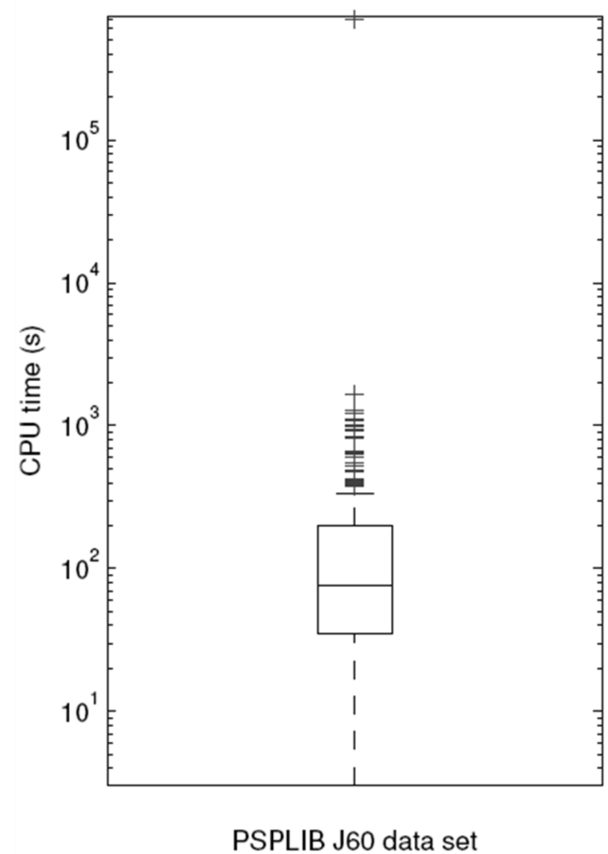
Stork (2001) was able to solve 179 out of 480 (37%) of the J30 problem instances. Even if activity durations have limited variability, we outperform Stork. In addition, we optimize over a class of policies that is more general!

Results: Computational performance (J60 - PSPLIB)

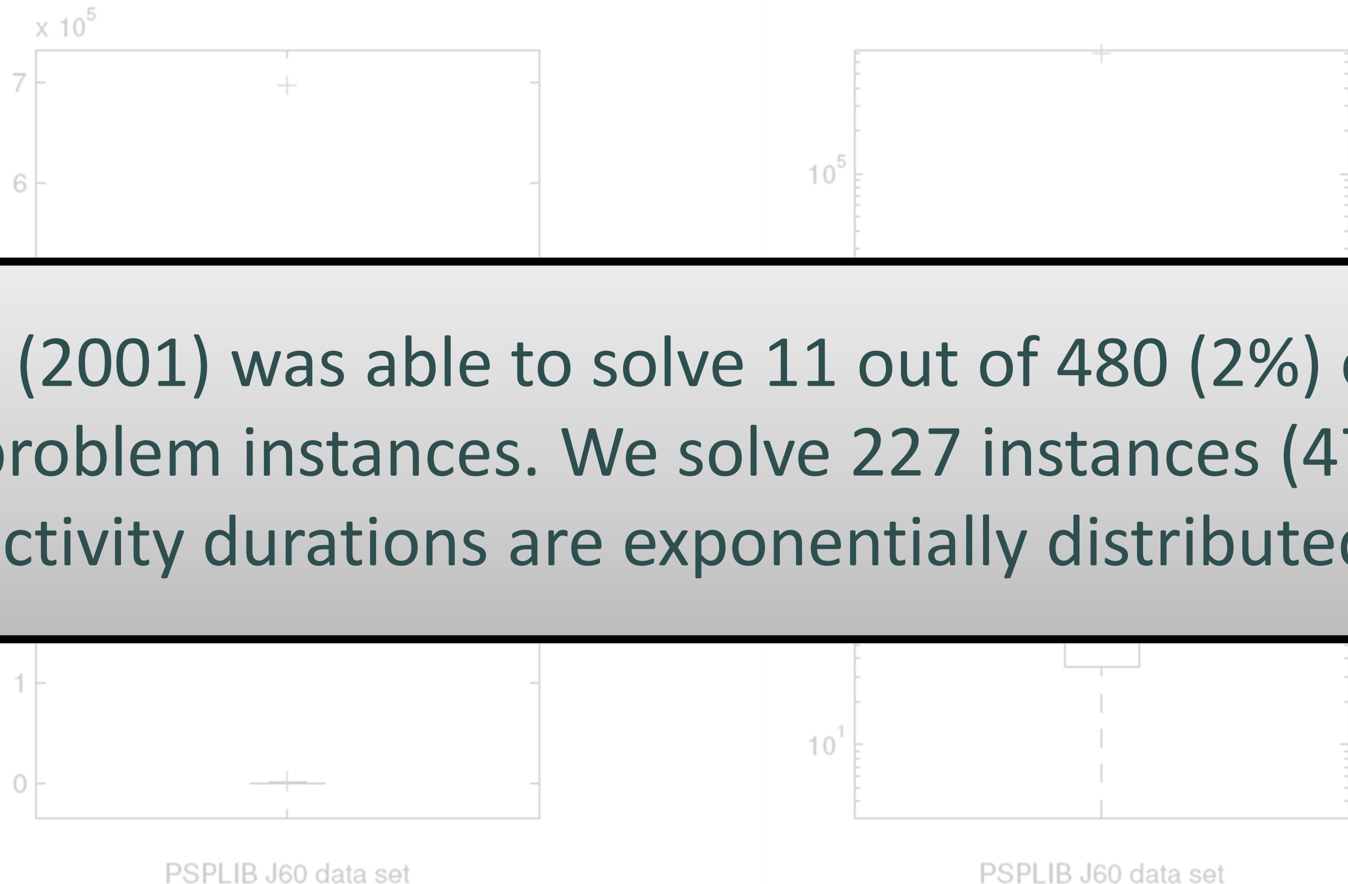


Results: Computational performance (J60 - PSPLIB)





Results: Computational performance (J60 - PSPLIB)



Stork (2001) was able to solve 11 out of 480 (2%) of the J60 problem instances. We solve 227 instances (47%) if activity durations are exponentially distributed.

Contributions

Contributions

We improve the SDP recursion of Creemers et al. (2010) and obtain an increase in computational efficiency of 5,000%.

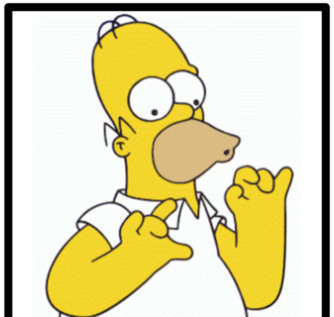
Contributions

We improve the SDP recursion of Creemers et al. (2010) and obtain an increase in computational efficiency of 5,000%.

We extend the model of Creemers et al. (2010) in order to solve the SRCPSP. We add resource constraints, general activity durations, and use a minimum-makespan objective.

Contributions

We improve the SDP recursion of Creemers et al. (2010) and obtain an increase in computational efficiency of 5,000%.



We extend the model of Creemers et al. (2010) in order to solve the SRCPSP. We add resource constraints, general activity durations, and use a minimum-makespan objective.

Solving the SRCPSP makes sense if activities have moderate- to high levels of duration variability. For this setting, our model outperforms the state-of-the art (both in solution quality & in computation speed).

?