Robust Multi-Agent Path Finding and Executing

prezentace - František Dostál

MFF UK

2022

Obsah

Motivace článku

Zavedení pojmu k-robustnosti a prokázání validity tohoto konceptu.

Konflikt

- a_i - agent a_i
- $\pi_i(t)$ - pozice agenta a_i v čase t
- $\langle a_i, a_j, t \rangle$ - koflikt agentů a_i, a_j v čase t
- **vertex** conflict - 2 agenti, 1 uzel v čase t
- **swapping** conflict -
  2 agenti, 1 hrana při přechodu z t-1 do t
- CBS - Conflict Based Search

CBS - Conflict Based Search

- low-level (A*) a high-level solver (CT)
- ve větvi spustíme low-level solver.
- identifikujeme konflikt
- vyřešíme konflikt (přidáme constraint do následníků)
- $\langle a_i, v, t \rangle$ zákaz agenta a_i na pozici v v čase t

Zpoždění

- $D = \langle a_i, t \rangle$ - zpoždění agent a_i, v přechodu z času t-1 do t
- **k-zpožděný konflikt** (k-delay conflict) - 2 agenti, 1 uzel v rozmezí k časových jednotek
- **plán neobsahuje k-zpožděný konflikt** $\Leftrightarrow$ **plán je k-robustní** (Pozorování 1)

Obsah

k-robustní MAPF

- kR-MAPF
- D - množina zpoždění
- plán π je robustní vůči $D \Leftrightarrow \pi$ je validní i po výskytu D
- **k-robustní plán** (k-robust) je plán robustní vůči $|D| \leq k$
- k-robustní plán je optimální pokud neexistuje k-robustní plán s menší cenou (cost)

Optimalita kR-MAPF

Lemma:
Pro všechna $k_1 < k_2$ platí, že cena optimálního k_1-robustního plánu je menší nebo rovna ceně optimálního k_2-robustního plánu.

$\Rightarrow$ trade-off robustnosti a ceny

$\Rightarrow$ k-robustnost je hard constraint

A* MAPF

- *n-agent state space*
  - stavy jsou rozmístění agentů na vrcholy grafu V.
- agent má akce pohybu a čekání
- akce je ve stavu aplikovatelná nebo ne ($\longrightarrow$ konflikt)

$\Rightarrow$ upravíme aplikovatelnost akcí.

A* kR-MAPF

To ovšem nezaručuje optimalitu!

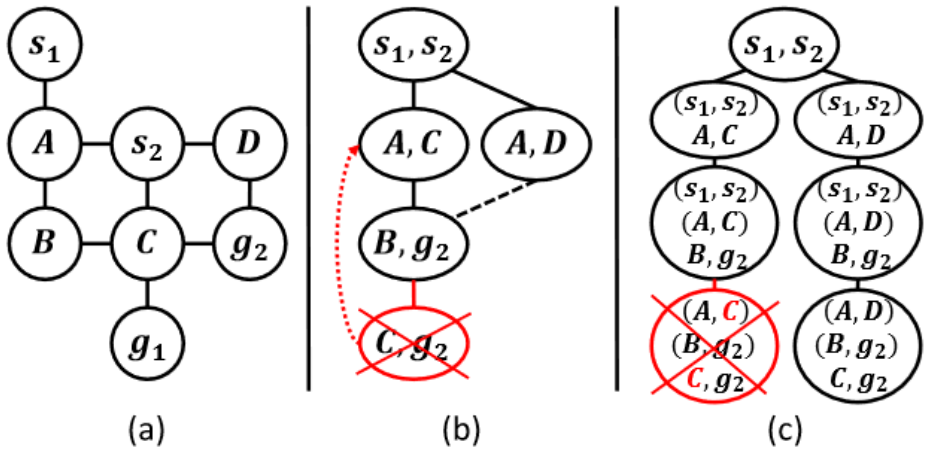


Figure 2: A MAPF problem (a) its search tree (b), and its k-robust search tree (c). The red lines show k-delay conflicts.

$\Rightarrow$ stav musí být určen i přes k provedených kroků

$\Rightarrow$ exponenciální růst stavového prostoru.

kR-CBS - Identifikace konfliktů

Z Pozorování 1 $\Rightarrow$ kontrola k-zpožděných konfliktů.

- delší kontrola konfliktu
- polynomiální vůči k

kR-CBS - Oprava konfliktů

- pro každý k-delay konflikt $\langle a_i, a_j, t \rangle \longrightarrow$ 2 uzly CT
- 1. uzel constraint a_j v $t + \Delta$
- 2. uzel constraint a_i v t
- sound, complete, optimal

IkR-CBS

- constraints v kR-CBS jsou slabší než by mohly být
- oprava konfliktů přidáním *range constraint* (intervalový constraint)
- **range constraint** $= \langle a_i, v, [t_1, t_2] \rangle$
- ne všechny range constraint musí zachovávat completeness a optimalitu

Definition 4.1 (Sound Range Constraints). A pair of range constraints for a given k-delay conflict is called *sound* iff every k-robust plan satisfies at least one of the constraints.

Proposition 4.1. If IkR-CBS resolves conflicts with sound pairs of range constraints then it is guaranteed to return an optimal k-robust plan if such exists.

IkR-CBS - Srovnání

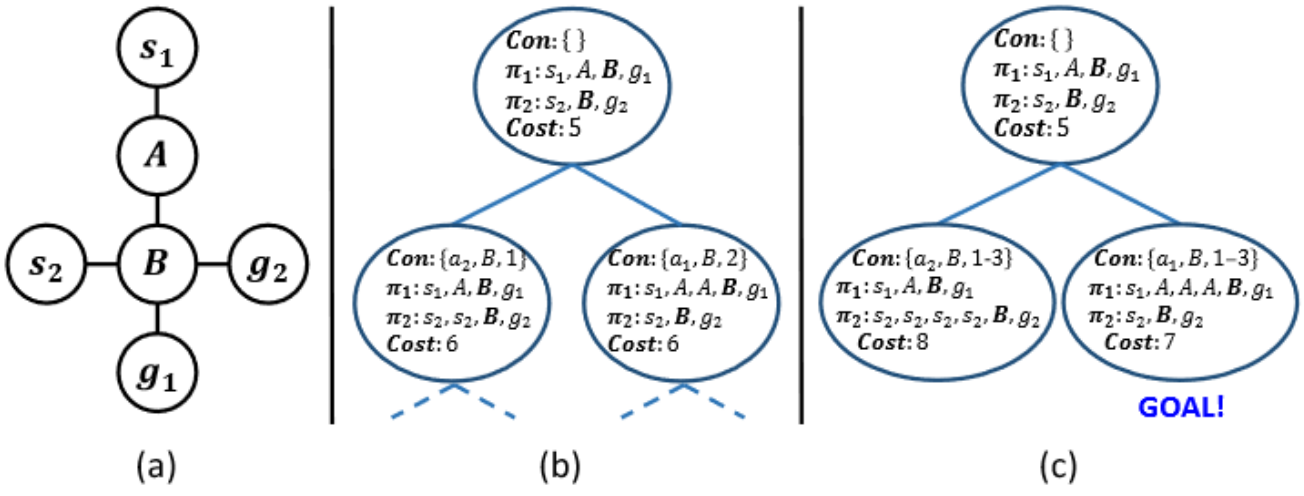


Figure 3: (a) The graph (b) The CT using the original time/vertex constraints (c) CT using the range constraints

lkR-CBS - hledání Sound RC

- symetrický pár constraints (SC) - oba agenti (uzly) mají zakázaný stejný interval $[t, t + k]$
- asymetrický pár constraints (AC) - zakázané intervaly mají různou délku
- AC - možná výpočetní výhoda
- kdo dostane kratší interval?

Corollary 4.2 (Symmetric range constraints). For any time step t, vertex v, and agents a_i and a_j, the range constraints $\langle a_i, v, [t, t + k]\rangle$, $\langle a_j, v, [t, t + k]\rangle$ are sound for solving a k-robust MAPF problem.

AC/SC

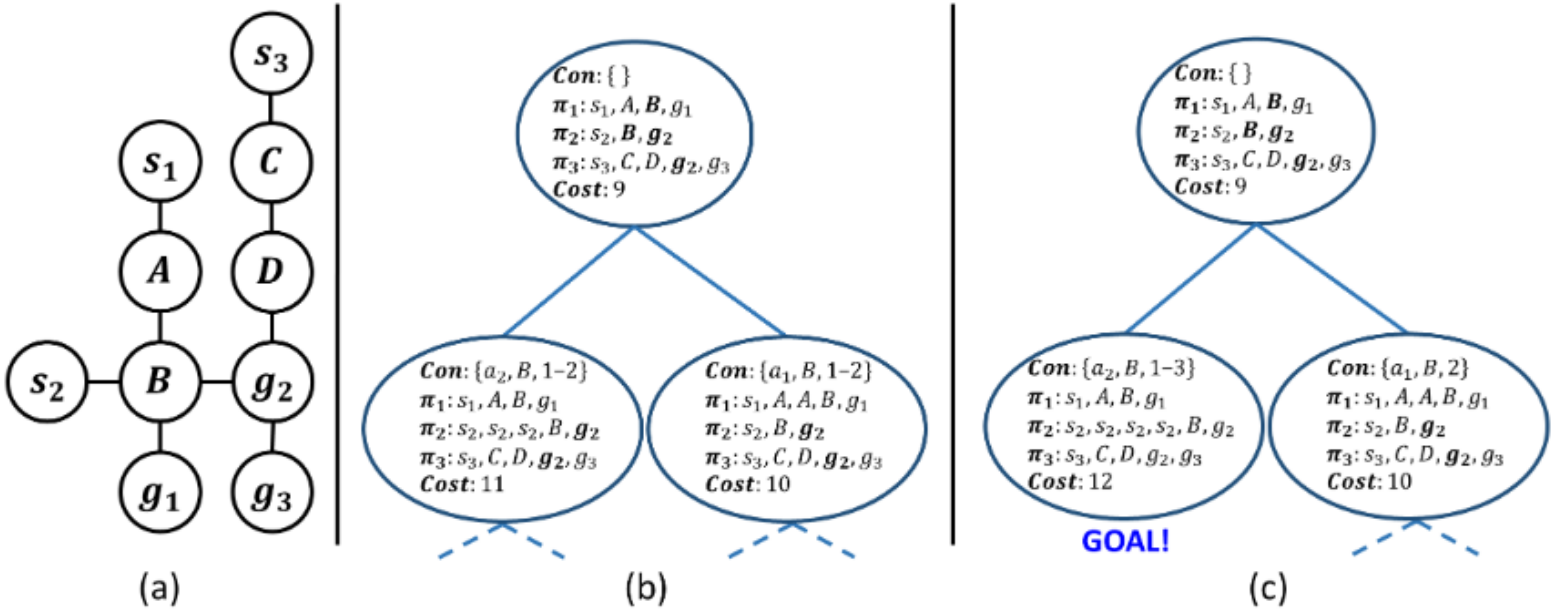


Figure 4: (a) The graph (b) The CT using the symmetric range constraints (c) CT using the asymmetric range constraints

SC(b) - 5 expanzí, AC(c) - 4 expanze

Obsah

Open Grid

- testy na mřížkách 8x8, 32x32, 64x64
- náhodně vybrané startovní a cílové body
- především srovnání upravených algoritmů
- nižší hustota agentů = stabilnější cena plánu

Open Grid - Výsledky

	Plan cost			Plan time (ms)						
				k=0	k=1			k=2		
n	k=0	k=1	k=2	All	KR	KR(A)	KR(S)	KR	KR(A)	KR(S)
4	22	22	22	6	17	**15**	16	229	130	**79**
6	31	31	32	5	25	26	**21**	1,087	416	**97**
8	40	41	43	7	29	24	**21**	2,535	1,007	**219**
10	49	51	54	42	166	126	**80**	22,986	7,261	**895**

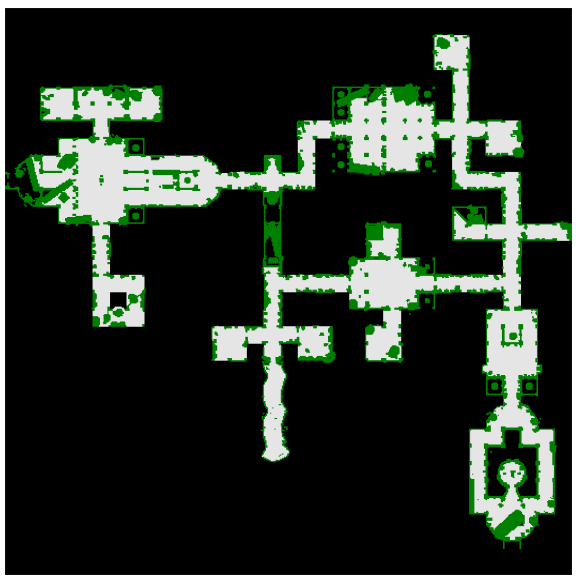
Table 1: Average plan cost (over all 50 instances) and planning runtime for different CBS-based k-robust solvers, on an 8x8 open 4-neighborhood grid.

	Plan cost			Plan time (ms)						
				k=0	k=1			k=2		
n	k=0	k=1	k=2	All	KR	IKR(A)	IKR(S)	KR	IKR(A)	IKR(S)
4	176	176	177	11	**10**	13	11	**11**	14	16
6	259	259	259	13	**17**	**17**	**17**	23	**22**	**22**
8	338	339	339	44	56	55	**54**	70	70	**69**
10	413	413	414	53	710	107	**68**	740	130	**81**

Table 2: Average plan cost and planning runtime for different CBS-based k-robust solvers, on an 64x64 open 4-neighborhood grid.

Dragon Age: Origins Mapa

- mapa z *movingai* repository
- náhodně vybrané startovní a cílové body
- 50 scénářů s 30 agenty, cesta nalezena pro 44 scénářů
- výsledné ceny se lišily minimálně - hodně prostoru
- potvrzuje výsledky předchozího experimentu



Zvyšování k na Grid domain

- mřížka 16x16
- srovnání výkonu při různých hodnotách k
- náhodně generovaných 50 problému pro daný počet agentů
- použit *IkR-CBS(S)* - z předchozích testů nejlepší

Zvyšování k - Výsledky

	Success rate							
n	k=0	k=1	k=2	k=3	k=4	k=5	k=6	k=7
5	50	50	50	50	47	47	45	41
10	50	50	46	43	34	29	21	15
15	49	45	35	19	13	8	3	1

	Cost							
n	k=0	k=1	k=2	k=3	k=4	k=5	k=6	k=7
5	49	49	49	49	50	50	50	51
10	116	116	117	118	122			
15	163	164	166					

	Time (ms)							
n	k=0	k=1	k=2	k=3	k=4	k=5	k=6	k=7
5	5	7	7	34	58	344	1,231	9,210
10	12	1,683	36,142	53,944	93,559			
15	17	8,805	38,678					

Table 3: Number of instances solved within the allocated time out, average cost, and average planning time, on an 16x16 open 4-neighborhood grid.

Warehouse Domain

- experiment převzat z článku *Multi-Agent Path Finding with Delay Probabilities*
- náhodně vybrané startovní a cílové body
- použit *IkR-CBS(S)*
- potvrzuje výsledky předchozího experimentu

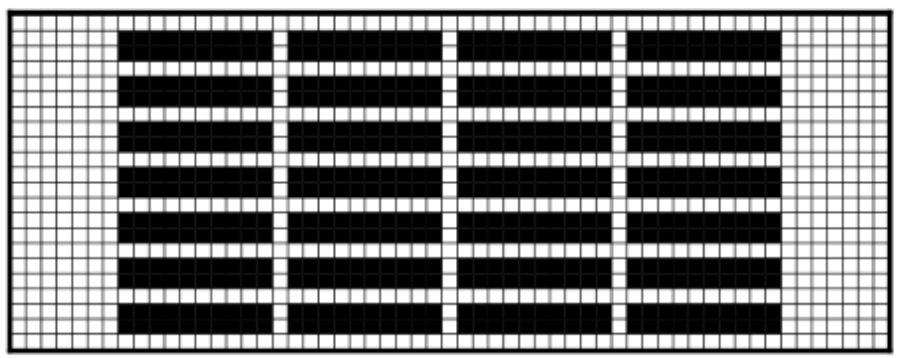


Figure 5: Warehouse domain, taken from Ma et al. (2017).

Warehouse Domain - Výsledky

	Cost			Time (ms)		
#Agents	k=0	k=1	k=2	k=0	k=1	k=2
5	148.43	148.54	148.71	110	110	116
10	297.33	297.80	298.58	1,042	1,176	1,570
15	435.76	436.80	438.58	1,816	2,058	12,746

Table 4: Average cost and planning runtime of kR-CBS on the warehouse domain.

Obsah

PICAT

- **Picat** - jazyk pro Constraint Programming
- problém lze převést na **SAT**, CP, MILP
- odkaz na model nefungoval... :(
- proměnná je určena trojicí agent, čas a pozice
- k-robustnost vynucená v rámci constraints

Obsah

Srovnání IkR-CBS vs. SAT

- Picat model zkompilován na SAT
- IkR-CBS používá symetrické RC
- optimalita obou algoritmů $\longrightarrow$ stejné ceny plánů

Grid 8x8

- srovnání času plánování
- 50 náhodných scénářů, 100% úspěšnost
- potvrzuje obecné předpoklady o přístupech založených na kompilaci

Obstacles	Cost		Planning time (ms)			
	k=1	k=2	k=1		k=2	
			CBS	Picat	CBS	Picat
12	35.16	36.30	1,511	**1,026**	**454**	1,805
16	39.24	40.68	4,142	**1,401**	6,572	**2,590**
19	39.67	42.47	3,362	**1,506**	8,727	**3,235**
22	34.20	36.91	6,979	**1,439**	14,042	**3,104**
25	28.26	29.52	9,834	**1,206**	12,957	**1,813**
32	16.28	18.73	**95**	322	1,488	**810**

Table 5: Solution cost and runtime results for 8x8 4-neighborhood grids with 6 agents, and a different number of obstacles.

Grid 32x32

- srovnání úspěšnosti
- 50 scénářů, 5 minutový limit řešení
- CBS citlivější na změny k a n

#Agents		10	15	20	25	30	35	40	45	50
k=1	Picat	**50**	**50**	**47**	**35**	15	2	0	0	0
	CBS	**50**	49	42	27	**22**	**7**	**1**	0	0
k=2	Picat	**50**	**49**	**38**	**9**	0	0	0	0	0
	CBS	**50**	45	31	6	**4**	0	0	0	0

Table 6: Number of instances solved within the allocated time out. The grids used are 32x32 4-neighborhood grids with 20% obstacles.

		Cost			Time		
#Agents		10	15	20	10	15	20
k=1	Picat	228.2	337.9	438.9	63.0	97.6	**111.3**
	CBS	228.2	337.9	438.9	**15.1**	**33.3**	249.8
k=2	Picat	228.9	339.7	441.6	88.6	173.8	**193.7**
	CBS	228.9	339.7	441.6	**15.3**	**107.4**	1,455.8

Table 7: Planning cost and planning time. The grids used are 32x32 grids with 20% obstacles.

Dragon Age Map

- zátěžový test
- Picat neschopný vyřešit jedinou instanci (min 5 agentů)

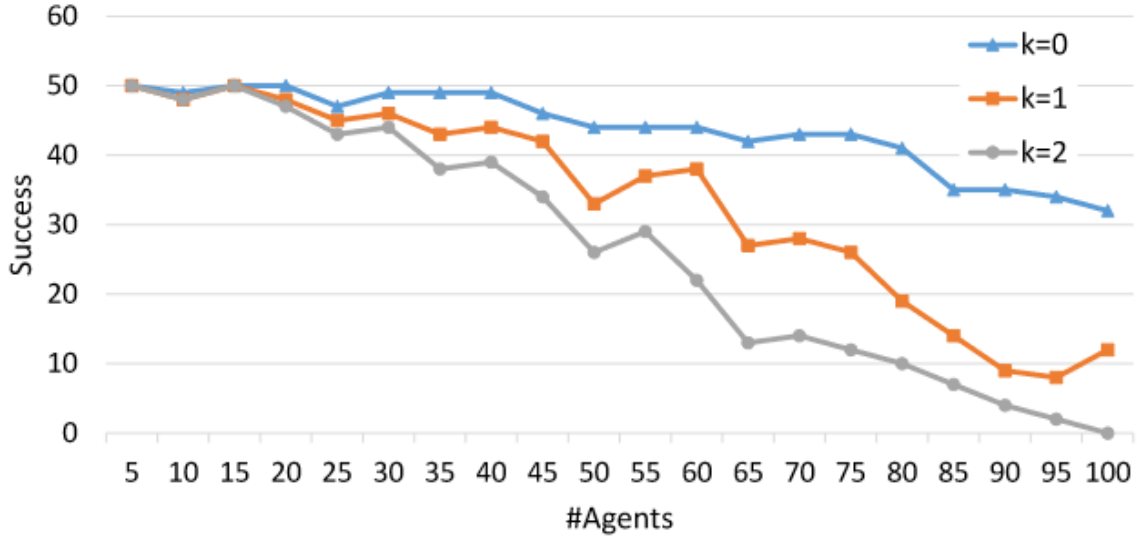


Figure 6: Success rate for kR-CBS over the `brc202d` DAO map.

Obsah

Execution policy (politka výkonu)

- učelem je reagovat na zpoždění v již prováděném plánu
- politka je robustní vůči plánu $\pi \Leftrightarrow$
  předejde všem kolizím během provádění plánu π
- zobecnění pojmu *robust plan-execution policy*
  (MAPF-DP, 1-robustní plánování)

Execution policy - kdy?

(1) Eager. Modify π immediately when a delay occurs.
(2) Reasonable. Modify π when a delay occurs but only if that delay is expected to cause a conflict later between the delayed agent and some other agent.
(3) Lazy. Modify π only when a conflict is expected to occur in the next time step.

Checking whether a conflict is expected to occur is done by simulating the execution of π from the current vertices of the agents. The logic behind "Reasonable" is that if a conflict is expected to occur then it is best to modify π to avoid it as soon as possible, while the logic behind "Lazy" is that future unexpected delays may resolve expected conflicts even without modifying π earlier.

Execution policy - jak a pro koho?

- **Replan.** Modify π by creating a completely new plan starting from the current time step using a MAPF solver.[6]
- **Repair.** Modify π by performing minor modifications to it.

Repair se pak dál dělí na:

- **All.** All agents that were not delayed in the current time step are forced to wait at the next time step.
- **Selective.** Some of the agents that were not delayed in the current time step are forced to wait at the next time step.

Execution policy - kombinace

- 9 kombinací
- Lazy repair ovšem nemusí dávat smysl
  spouští se až při konfliktu a zpoždění už mohla proběhnout
- Lazy repair nebyla implementována, Selektivní politky nebyly analyzovány
- předpoklad rozpoznání konfliktu před pohybem
- MCP - Minimal Communication Policy

Execution policy - experiment 1.

- experiment s náhodně vynuceným zpožděním
- pravděpodobnost $p = 0.1, 0.01, 0.001,$
- síť 8x8, 20 agentů
- 0-robustní plány

Execution policy - výsledky 1.

	Cost	#Mod.	Time (ms)	Cost	#Mod.	Time (ms)	Cost	#Mod.	Time (ms)
	Delay probability = 0.001			Delay probability = 0.01			Delay probability = 0.1		
Eager All	111.08	0.26	**0**	128.33	1.99	**0**	521.24	27.17	**0**
Reasonable All	108.51	**0.06**	**0**	117.20	**0.48**	**0**	282.93	9.11	**0**
MCP	108.34	0.21	**0**	111.08	2.79	**0**	127.88	7.38	**0**
Eager replan	**108.10**	0.2	14	**108.28**	1.04	254	**111.17**	26.47	27,313
Reasonable replan	108.18	0.08	16	108.56	0.93	975	115.76	11.75	11,332
Lazy replan	108.41	0.08	3	110.22	0.87	387	125.94	**3.62**	2,471

Table 8: Comparison of the different replanning polices.

Execution policy - experiment 2.

- měříme vliv robustnosti původního plánu
- pravděpodobnost $p = 0.1$
- síť 16x16
- pouze MCP - balanc ceny, počtu a času přeplánování

Execution policy - výsledky 2.

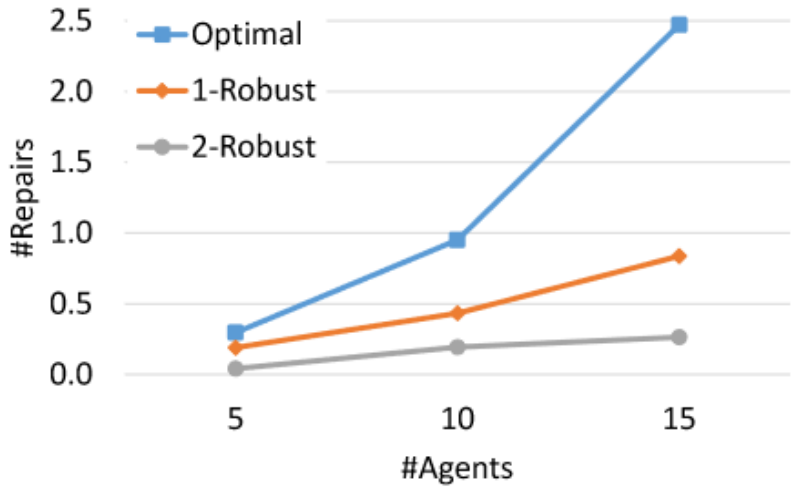


Figure 8: kR-CBS for MCP on the instances succeeded in Table 3 on a 16x16 grid

Obsah

Závěr

- propozice k-robustního plánování a několika metod řešení
- Picat je vhodný pro malé sítě s vysokou hustotou populace.
- CBS vhodný pro velké, řídké mapy.
- k-robustní plán lze najít v realistickém čase
- k-robustní plán výrazně snižuje nutnost online přeplánování
- otevřené problémy pro CBS algorimtus, ICTS, různé k pro různé agenty etc.

Zpožděná komunikace

- každých T kroků se agenti informují o svém stavu/zpoždění
- k-robustnost je "částečně" užitečná
- podobně jako EP rozlišuje se eager, reasonable, lazy, T-replan, T-repair

- **T-All.** Let Δ_i be the number of delays agent i experienced in the last T time steps and let Δ_{max} be the maximum over all Δ_i. In a T-All action, each agent needs to wait $\Delta_{max} - \Delta_i$ time steps.

- **T-Selective.** This plan repair method is an iterative process that continues as long as there is a T-delay conflict in the current plan π. Let $\langle a_i, a_j, t \rangle$ be such a conflict, and and let $\Delta \leq T$ be the Δ for which $\pi_i(t) = \pi_j(t + \Delta)$. The T-selective plan repair action will force agent a_j to wait $T - \Delta$ time steps in its current vertex, thereby resolving the corresponding T-delay conflict.

Zpožděná komunikace - robustnost

Theorem 7.1. *Eager T-replan, reasonable T-replan, and lazy T-replan, are robust for a given communication frequency T w.r.t to an initial plan that is T-robust.*

Theorem 7.2. *Eager T-All, Reasonable T-All, and Reasonable T-Selective, are robust for a given communication frequency $T > 0$ w.r.t to an initial plan that is T-robust.*

Zpožděná komunikace - výsledky

	Cost				Time (ms)			
Delay probability = 0.1								
	T=0	T=2	T=4	T=6	T=0	T=2	T=4	T=6
Eager T-All	89.53	72.12	69.58	73.61	0.00	0.00	0.00	0.00
Reasonable T-All	53.86	56.63	60.43	66.48	0.00	0.00	0.00	0.00
Reasonable T-Selective	**48.72**	53.03	58.78	66.00	0.00	0.00	0.00	0.00
MCP	**48.72**	**52.31**	**58.74**	**65.80**	0.00	0.00	0.00	0.00
Eager T-replan	47.57	52.37	58.65	**62.96**	9.06	13.30	12.36	11.84
Reasonable T-replan	**47.53**	53.12	58.25	65.85	7.23	9.57	5.68	3.51
Lazy T-replan	47.57	**51.94**	**57.99**	65.74	6.07	10.09	11.06	9.64
Delay probability = 0.2								
	T=0	T=2	T=4	T=6	T=0	T=2	T=4	T=6
Eager T-All	220.18	95.56	81.94	84.92	0.00	0.00	0.00	0.00
Reasonable T-All	67.67	68.22	70.33	74.59	0.00	0.00	0.00	0.00
Reasonable T-Selective	**55.64**	60.55	66.01	72.76	0.00	0.00	0.00	0.00
MCP	55.78	**58.67**	**65.30**	**72.44**	0.00	0.00	0.00	0.00
Eager T-replan	**53.23**	58.86	64.60	**69.31**	24.91	25.42	25.19	24.26
Reasonable T-replan	53.35	60.01	64.45	78.91	11.58	16.16	19.80	15.36
Lazy T-replan	53.57	**58.63**	**64.19**	71.11	7.08	28.05	37.57	25.58
Delay probability = 0.3								
	T=0	T=2	T=4	T=6	T=0	T=2	T=4	T=6
Eager T-All	651.97	122.80	101.04	100.55	0.00	0.00	0.00	0.00
Reasonable T-All	86.62	84.53	82.72	86.81	0.00	0.00	0.00	0.00
Reasonable T-Selective	65.04	69.28	75.15	81.79	0.00	0.00	0.00	0.00
MCP	**64.75**	**68.80**	**73.30**	**81.25**	0.00	0.00	0.00	0.00
Eager T-replan	**60.99**	67.73	**71.52**	**76.62**	29.23	37.23	40.85	41.47
Reasonable T-replan	61.14	68.08	72.17	80.16	15.23	23.77	25.59	37.44
Lazy T-replan	61.31	**66.91**	71.55	80.46	14.43	60.42	63.41	62.77

Table 9: Different replanning polices on instances with 10 agents.