AI Planning – How to plan planning

Filip Dvořák

Contents

- Introduction
- Motivation
- State of the field

Plánování

- V modelu světa se snažíme najít plán, který z počátečního stavu světa dosáhne cíle.

What can we plan?

- Anything
  - But, …

- Variants of AI planning
  - Observability (partially, fully)
  - Uncertainty
  - Result expectance (realtime, dynamic, offline)
  - Quality expectance (satisficing, optimal)
  - Human interaction (assisted, automated)

STRIPS formalism

- (V,I,G,O)
  - V – set of atoms
  - I – set of initial literals
  - G – set of goal literals
  - O – set of operators
    - open(person x, door y, key z){
      pre: closed(y), has(x,z), nearby(x,y)
      eff^+: open(y)
      eff^-: closed(y)

Motivation

- Domain independence
  - Reusability
  - Flexibility
  - Performance (!)

Greenhouse Logistic

- Phenomics facilities
  - Accelerated growth of genetically modified plants.
- Completely automated environment
- State space
  - 10^{738} (University of Adelaide)

State of the field

- Optimality?
  - Too expensive (at least guaranteed optimality).
  - Not useful for practice (scaling).
    - We are still in PSPACE …
- Satisfaction & sufficiency = Satisficing planning

- Domain “independent” structural analysis
  - Heuristics

Search

- Forward search
- Backward search
- Representation dependant
  - POCL
  - BlackBox (CSP, SAT, Evolutionary)

Heuristics

- h: $2^{|V|}$ -> R
  - $h^*(s)$ being optimal
- Admissible heuristic
  - $h(s) \leq h^*(s)$
- h dominates h'
  - $h'(s) \leq h(s)$

Heuristics

- Delete relaxation
  - h^{+},h^{max},h^{add},h^{FF},h^{pmax},h^{sa}
- Critical paths
  - h^{m}
- Abstractions
  - Pattern databases, merge-and-shrink, structural patterns
- Landmarks
  - h^{LM}, h^{L}, h^{LA}

Combining heuristics

- Maximum, sum
- Cost alternation
- Alternation
- Pareto optimality

References

- **Strengthening Landmark Heuristics via Hitting Sets**. In Helder Coelho, Rudi Studer and Michael Wooldridge (eds.), Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010), pp. 329-334. IOS Press 2010.
- **Sound and Complete Landmarks for And/Or Graphs**. In Helder Coelho, Rudi Studer and Michael Wooldridge (eds.), Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010), pp. 335-340. IOS Press 2010.
- **The Scanalyzer Domain: Greenhouse Logistics as a Planning Problem**. In Ronen Brafman, Héctor Geffner, Jörg Hoffmann and Henry Kautz (eds.), Proceedings of the 20th International Conference on Automated Planning and Scheduling (ICAPS 2010), pp. 234-237. AAAI Press 2010.
- **The More, the Merrier: Combining Heuristic Estimators for Satisficing Planning**.In Ronen Brafman, Héctor Geffner, Jörg Hoffmann and Henry Kautz (eds.), Proceedings of the 20th International Conference on Automated Planning and Scheduling (ICAPS 2010), pp. 246-249. AAAI Press 2010.
- **Landmarks, Critical Paths and Abstractions: What's the Difference Anyway?** In Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPS 2009), pp. 162-169. AAAI Press 2009.
- **Preferred Operators and Deferred Evaluation in Satisficing Planning**. In Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPS 2009), pp. 273-280. AAAI Press 2009.

Filuta
Plánování s časem a zdroji

Filip Dvořák, Roman Barták

Matematicko-fyzikální fakulta Univerzity Karlovy

Plánování s časem a zdroji

- V modelu světa se snažíme najít plán, který z počátečního stavu světa dosáhne cíle.
- Plánování s explicitním časem.
  - Akce mají dobu trvání, jsou rozvrženy v čase a jejich efekty mohou interferovat.
- Proč zdroje v plánování?
  - Reálné problémy je vyžadují.
  - Přenesením zdrojů do plánování můžeme sledovat kritéria zajímavá pro rozvrhování již během konstrukce plánu.

Filuta plánuje …

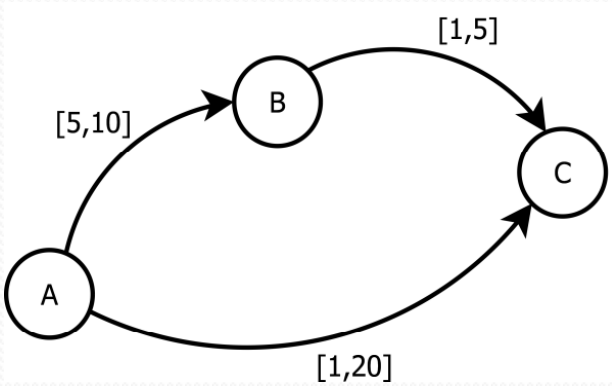
- Offline.
- V plně pozorovatelném a deterministickém světě.
- Minimalizuje čas nutný k vykonání plánu.
- Vstup je v PDDL (verze 2.1: typing, durative-actions, částečně numeric-fluents).

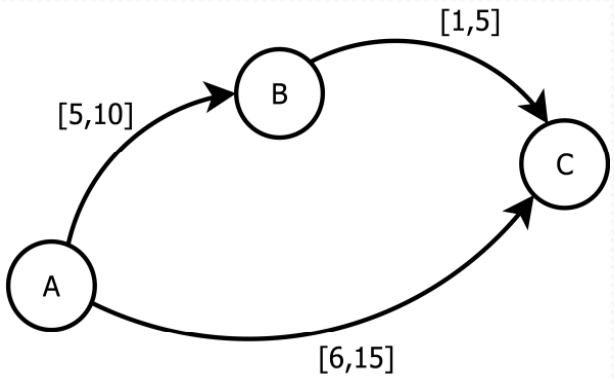
- Navážeme popisem reprezentace času, světa, zdrojů a akcí.

Reprezentace času

- Simple Temporal Problem (~CSP)
  - Množina časových proměnných, reprezentujících události.
  - Množina binárních podmínek (A [5,10] B)
- Proměnné a podmínky tvoří časovou síť (STN)
  - Síť lze minimalizovat a odvodit tak mezní podmínky mezi všemi páry proměnných.
  - Incrementálně v $O(n^2)$, kde n je počet proměnných.

Simple Temporal Network

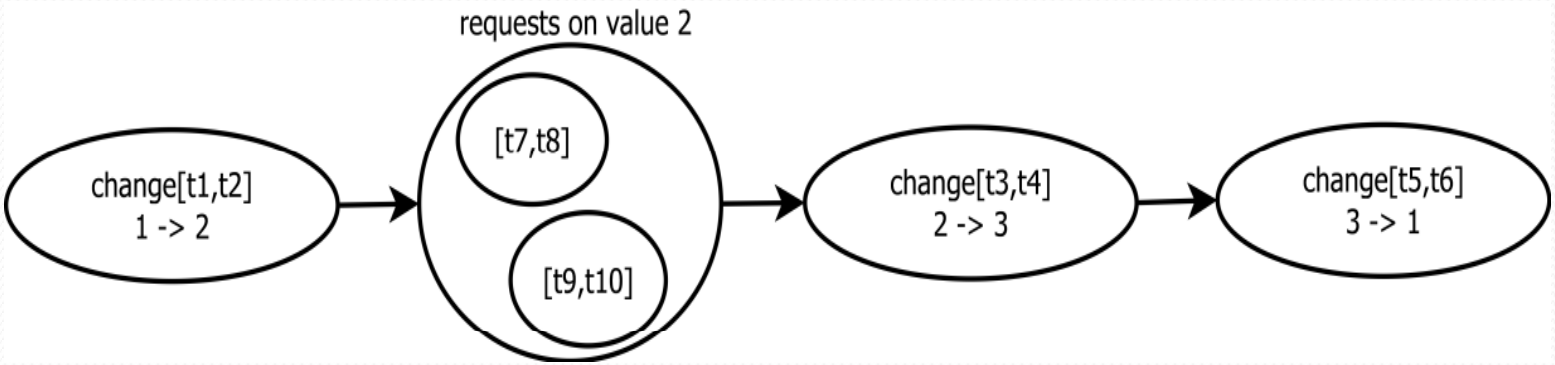




Reprezentace světa

- Stavové proměnné
  - Množiny vzájemně výlučných výroků o světě.
  - Stav světa je popsán jako přiřazení hodnot proměnným.
- Časové databáze
  - Pro každou stavovou proměnnou vytvoříme databázi.
  - Databáze modeluje posloupnost změn hodnot stavové proměnné v čase.
  - Úvodní hodnoty v časových databázích kódují počáteční stav světa.
  - Změny jsou v databázi anotovány časovými proměnnými.

Časové databáze



Reprezentace zdrojů

- Modelujeme známé unární zdroje, diskrétní zdroje a rezervoáry.
- Pro každý zdroj v plánovacím problému vytvoříme instanci zdroje.
  - Instance zdroje je množina zdrojových událostí.
  - Události jsou anotovány časovými proměnnými.
  - Např. „dveře jsou otevřeny v čase X až Y“.
- Zdroje spravujeme jako problém omezujících podmínek nad časovou sítí.

Reprezentace akcí

- Časové operátory.
- Parametrizovány časovými proměnnými.
- Obsahují změny a požadavky na hodnoty
- Příklad:
  - přesuň_auto_A_B(t_s,t_e){
    $t_e - t_s = 22$,
    pozice-auto[t_s,t_e]: A→B,
    řidič[t_s,t_e]: auto,
    palivo-auto[t_e,t_e]:-10
    }

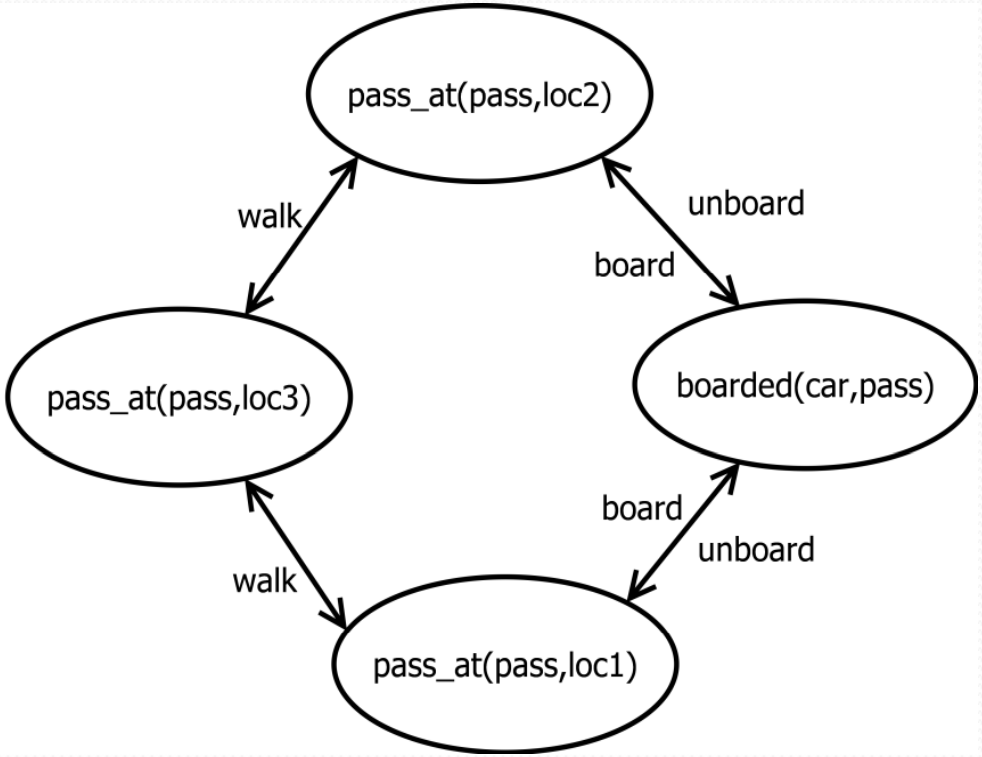
Plánovací problém

- (STN, TDBs, Actions, RIs, Goals)
- Počáteční stav
  - (STN, TDBs, RIs, Plán)
    - Plán je prázdný.
- Cílový stav
  - (STN', TDBs', RIs', Plán')
    - STN', TDBs', RIs' jsou konzistentní a vyvinuly se z STN, TDBs, RIs prostřednictvím přidávání akcí do Plán.
    - Cíle jsou splněny.

Předzpracování

- Grafy doménových přechodů
  - Nad doménami stavových proměnných.
  - Orientovaný (multi) graf, vrcholy odpovídají hodnotám stavové proměnné, hrana (v,w) odpovídá akci obsahující změnu stavové proměnné v→w.
  - Najdeme všechny nejkratší cesty.
    - Heuristika pro výběr akce (#hran, čas).
    - Dolní mez pro prořezávání (čas).

Grafy doménových přechodů



Algoritmus

- Plánovací problém rozdělíme na podproblémy dle cílů
  - Cyklicky řešíme podproblémy, dokud nejsou všechny cíle splněny.
- Podproblémy řešíme metodou větví a mezí
  - Větvíme na
    - Výběru hrany v doménovém grafu (= volba akce).
    - Výběru časového kontextu pro změnu nebo požadavek.
    - Výběru akce řešící zdrojový konflikt.
  - Meze
    - Horní mez je dosud nejlepší nalezené řešení.
    - Dolní mez je spočtena ze všech zbývajících cest v doménových grafech k aktuálnímu stavu.

Algoritmus - prohledávání

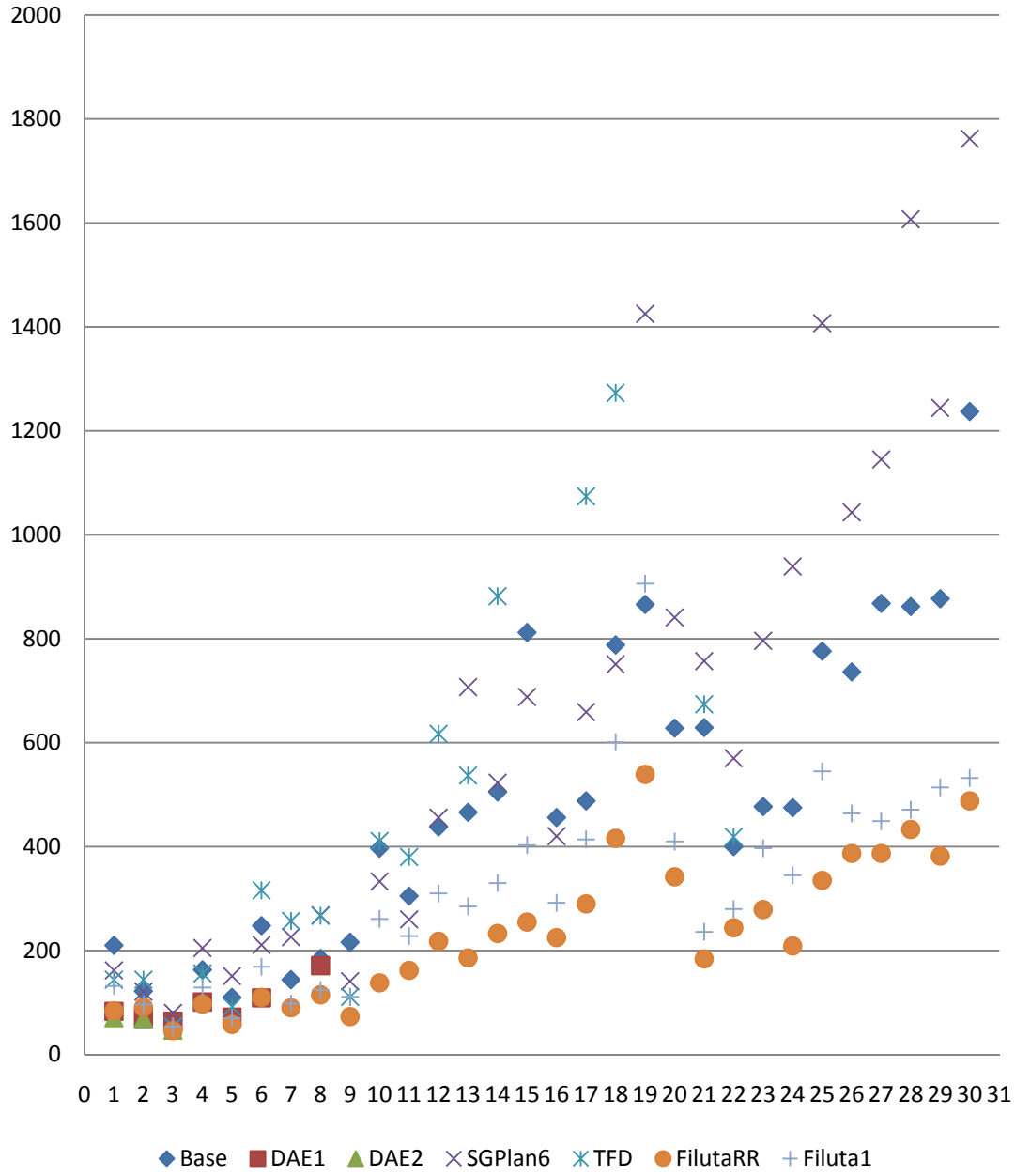
- Vyřešení podproblému (splnění jednoho cíle) představuje nalezení cesty v odpovídajícím doménovém grafu.
  - Prohledáváme do hloubky s heuristikami (#hran, čas) a fewest-options-first.
  - Každý přechod hrany vytvoří nutnost uspokojit ostatní změny, požadavky a zdrojové události, které obsahovala akce reprezentující tuto hranu.
    - Uspokojení změn a požadavků je opět problém nalezení cesty v (jiném) grafu.
    - Zdrojová událost může vyvolat zdrojový konflikt, který je nutno řešit přidáním další akce.

Algoritmus – anytime verze

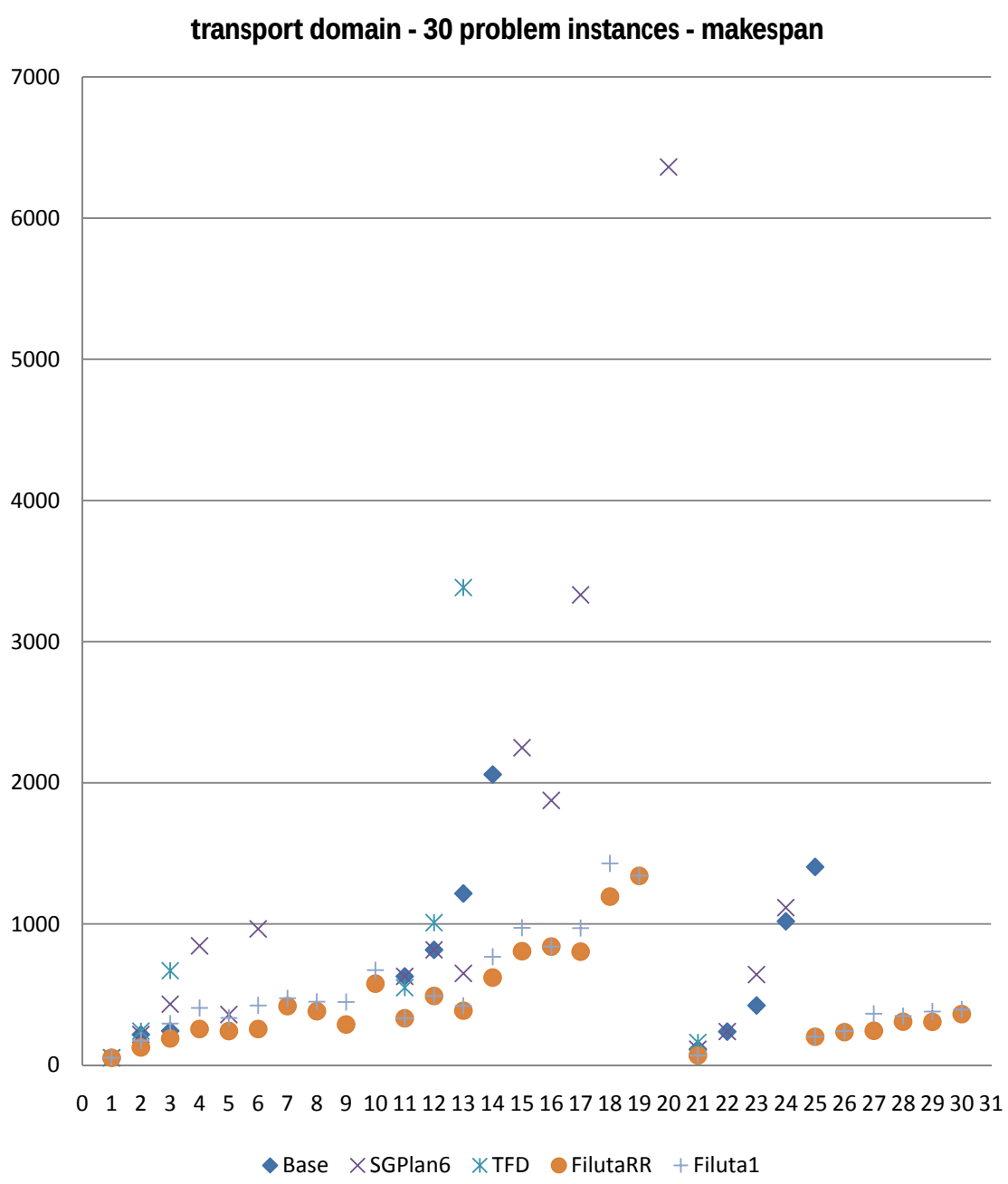
- Popsaný algoritmus skončí po nalezení prvního plánu.
- Rozšíříme algoritmus o náhodné restarty pořadí, v jakém jsou splňovány cíle.
- Zároveň každý restart používá předchozí řešení jako prvotní horní mez.

Jak dobře to funguje?

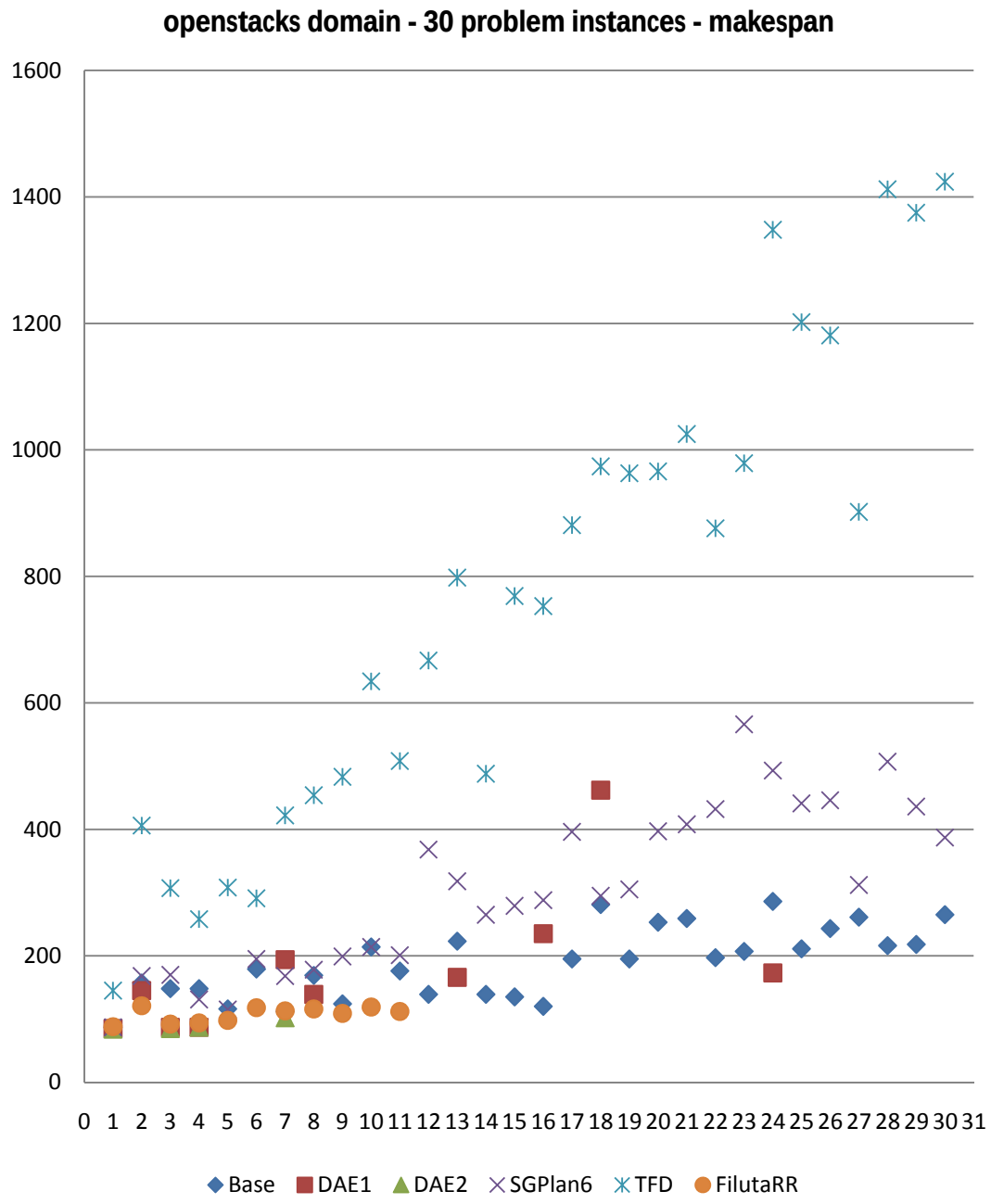
- 3 domény z IPC2008, 30 problémů v každé
  - Elevators: makespan v průměru o 38% nižší,
  - Transport: makespan v průměru o 50% nižší,
  - Openstacks: makespan v průměru o 13% nižší.
- Celkem vyřešeno
  - Elevators: 30/30,
  - Transport: 26/30,
  - Openstacks: 11/30.



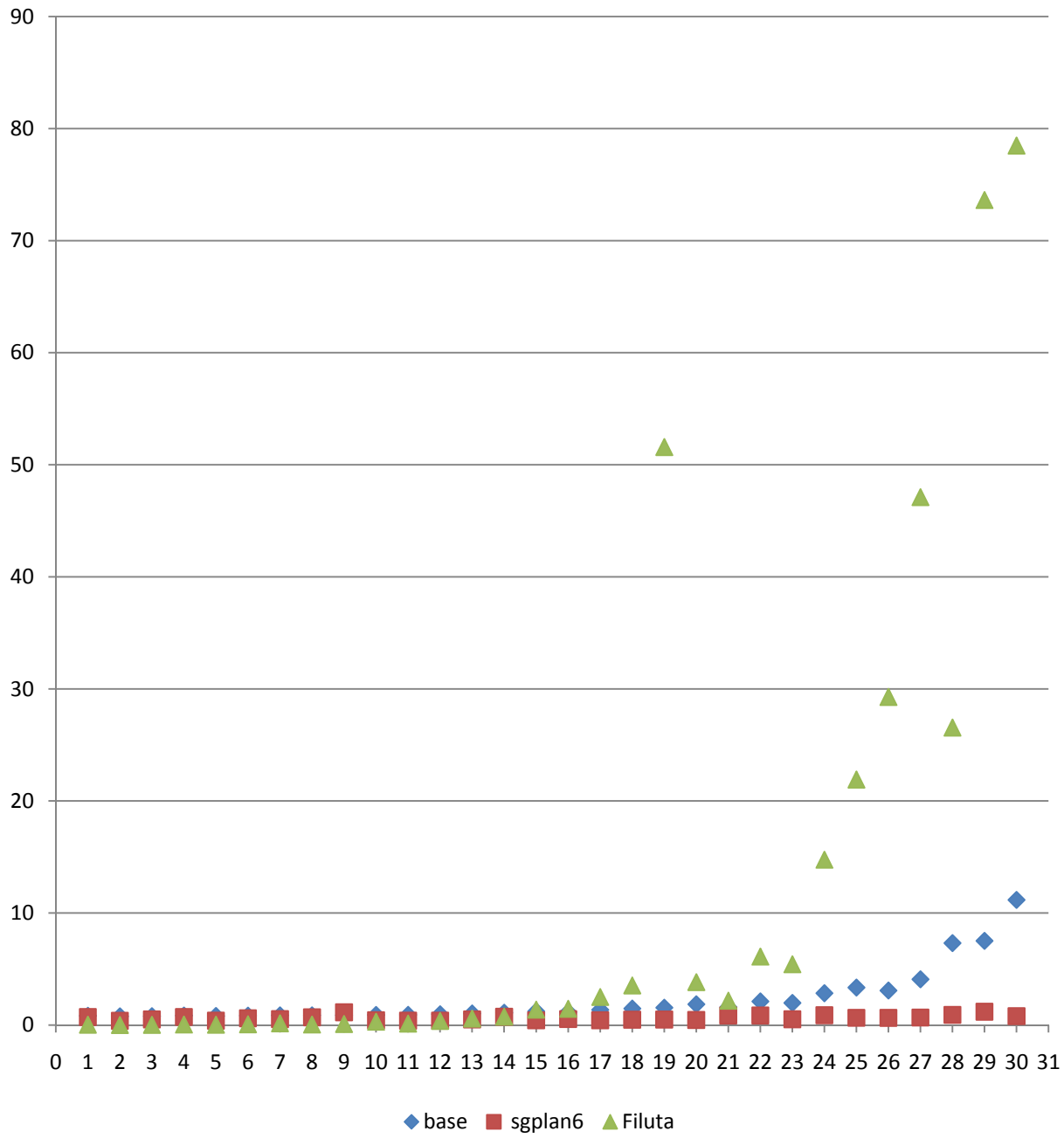
elevators domain - 30 problem instances - makespan							
	Base	DAE1	DAE2	SGPlan6	TFD	Filuta^{RR}	Filuta^{1}
1	210	83	**71**	162	144	84	132
2	122	71	**69**	121	144	91	96
3	66	64	47	80	54	**46**	54
4	163	101		205	156	**97**	129
5	110	72		151	92	**58**	70
6	248	**109**		211	316	110	169
7	144			226	257	**90**	98
8	185	171		268	267	**115**	124
9	216			141	111	**73**	111
10	397			333	411	**138**	261
11	305			260	380	**162**	228
12	438			456	617	**218**	310
13	466			707	537	**186**	285
14	505			523	882	**233**	330
15	812			688		**255**	403
16	456			420		**225**	292
17	488			659	1074	**290**	414
18	788			751	1273	**416**	601
19	866			1425		**539**	906
20	628			841		**342**	410
21	629			757	674	**184**	236
22	400			570	419	**244**	280
23	477			796		**279**	397
24	475			939		**209**	345
25	776			1407		**335**	545
26	736			1043		**387**	464
27	868			1145		**387**	449
28	862			1607		**433**	471
29	877			1244		**382**	514
30	1237			1762		**488**	532



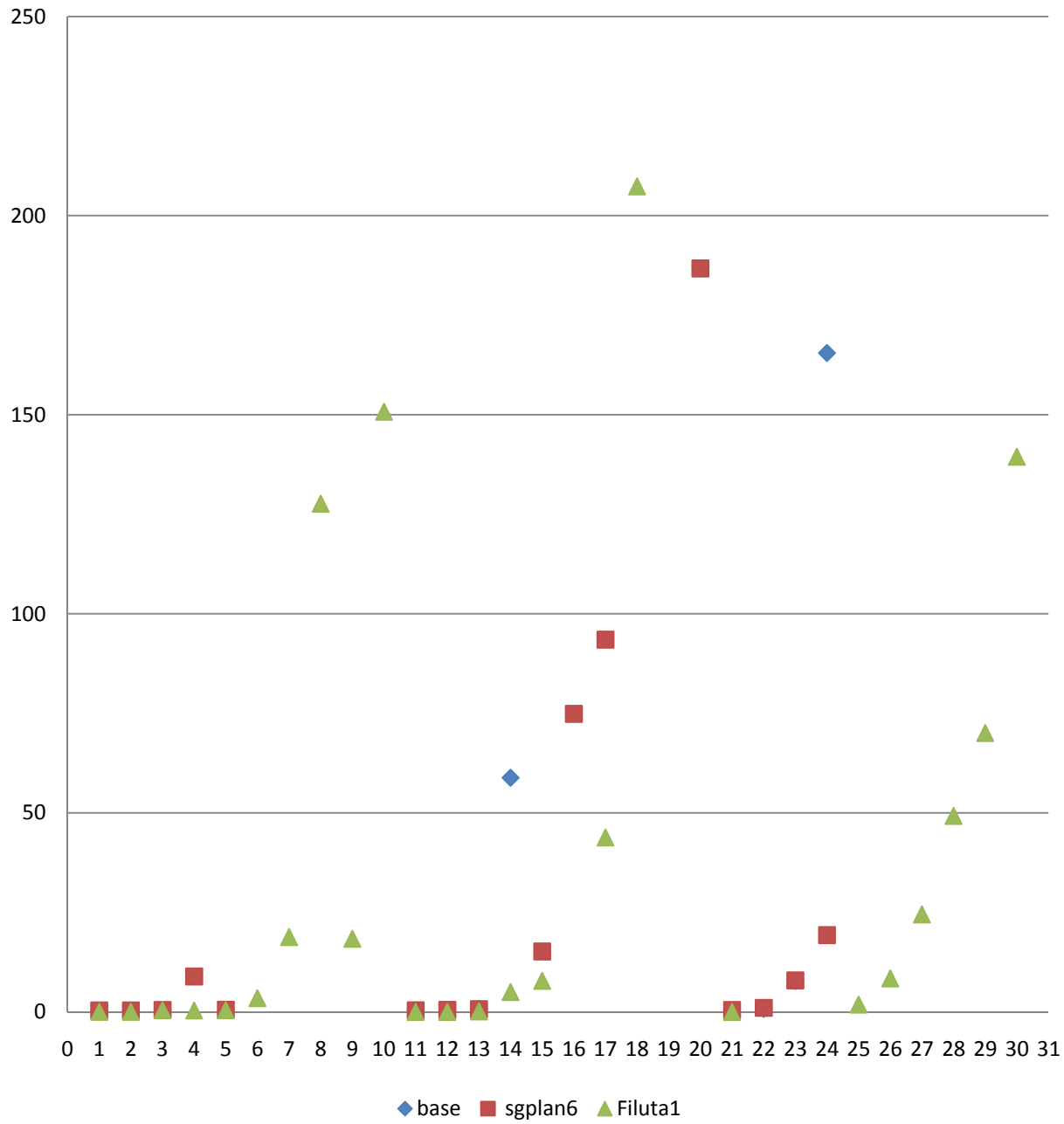
transport domain - 30 problem instance - makespan					
	Base	SGPlan6	TFD	**Filuta^{RR}**	**Filuta^{1}**
1	**52**	**52**	**52**	**52**	52
2	217	217	241	**126**	173
3	243	432	669	**189**	295
4		845		**256**	405
5		359		**242**	335
6		965		**256**	423
7				**418**	474
8				**382**	449
9				**288**	447
10				**577**	673
11	629	629	549	**332**	332
12	817	817	1009	**490**	490
13	1216	650	3383	**386**	420
14	2059			**620**	768
15		2249		**807**	973
16		1875		**840**	840
17		3331		**804**	971
18				**1194**	1429
19				**1341**	1341
20		**6362**			
21	113	113	161	**69**	69
22	**238**	**238**			
23	**423**	642			
24	**1019**	1116			
25	1404			**201**	201
26				**234**	241
27				**244**	364
28				**308**	348
29				**307**	380
30				**362**	394



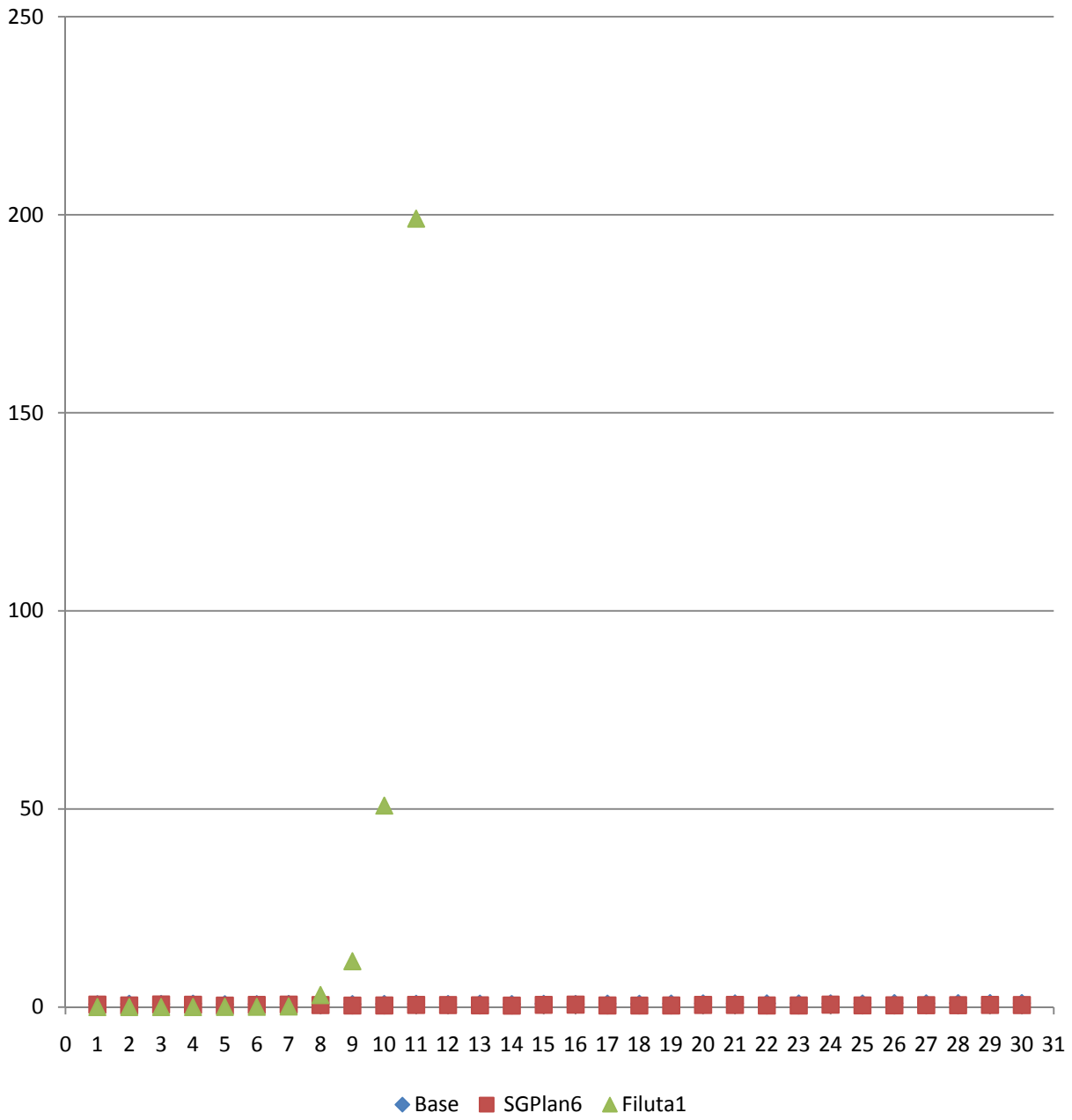
openstacks domain - 30 problems - makespan						
	Base	DAE1	DAE2	SGPlan6	TFD	Filuta^{RR}
1	87	85	**84**	87	145	88
2	157	145		168	406	**121**
3	148	87	**85**	170	307	92
4	148	**87**	**87**	131	258	94
5	116			115	308	**98**
6	179			195	291	**118**
7	112	194	**102**	168	422	113
8	169	139		178	454	**116**
9	124			199	483	**109**
10	214			214	634	**119**
11	176			201	508	**112**
12	**139**			368	667	
13	223	**166**		318	798	
14	**139**			265	488	
15	**135**			279	769	
16	**120**	235		288	753	
17	**195**			396	881	
18	**281**	462		295	974	
19	**195**			305	963	
20	**253**			397	966	
21	**259**			408	1025	
22	**197**			432	876	
23	**207**			566	979	
24	286	**173**		493	1348	
25	**211**			441	1202	
26	**243**			446	1181	
27	**261**			312	902	
28	**216**			507	1412	
29	**218**			436	1375	
30	**265**			387	1424	



elevators - time (s)			
	base	sgplan6	**Filuta[1]**
1	0.836	0.716	0.031
2	0.784	0.384	0.001
3	0.796	0.508	0.016
4	0.856	0.724	0.047
5	0.816	0.392	0.031
6	0.848	0.608	0.062
7	0.876	0.532	0.156
8	0.876	0.696	0.047
9	0.884	1.144	0.094
10	0.908	0.432	0.297
11	0.92	0.396	0.125
12	0.976	0.4	0.361
13	1.044	0.508	0.578
14	1.112	0.744	0.751
15	1.228	0.424	1.375
16	1.16	0.536	1.453
17	1.364	0.432	2.502
18	1.48	0.484	3.532
19	1.556	0.496	51.579
20	1.864	0.452	3.828
21	1.608	0.852	2.172
22	2.116	0.856	6.109
23	1.98	0.516	5.422
24	2.848	0.884	14.751
25	3.352	0.648	21.907
26	3.088	0.64	29.281
27	4.088	0.676	47.109
28	7.32	0.912	26.546
29	7.516	1.192	73.625
30	11.173	0.8	78.485

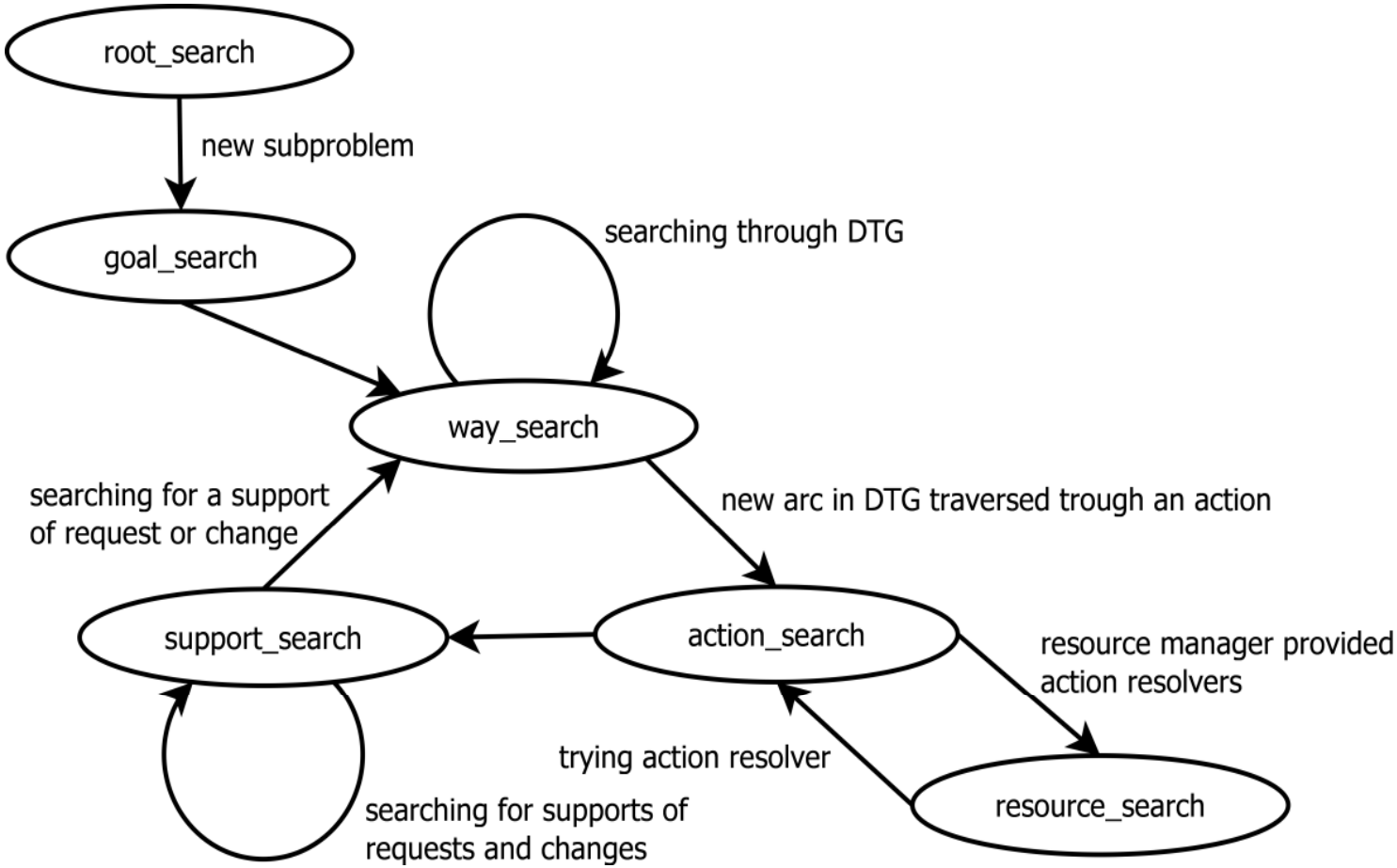


transport - time (s)			
	base	sgplan6	**Filuta[1]**
1	0.248	0.388	0.031
2	0.296	0.392	0.031
3	0.392	0.5	0.468
4		8.909	0.375
5		0.572	0.454
6		478.698	3.4693
7			18.828
8			127.656
9			18.406
10			150.734
11	0.408	0.416	0.001
12	0.4	0.508	0.016
13	0.772	0.688	0.157
14	58.84		5.016
15		15.181	7.828
16		74.869	1194.719
17		93.502	43.828
18			207.343
19			1647.611
20		186.732	
21	0.368	0.488	0.001
22	0.8	1.024	
23	7.804	7.936	
24	165.518	19.293	
25	460.069		1.875
26			8.437
27			24.516
28			49.251
29			70.062
30			139.453

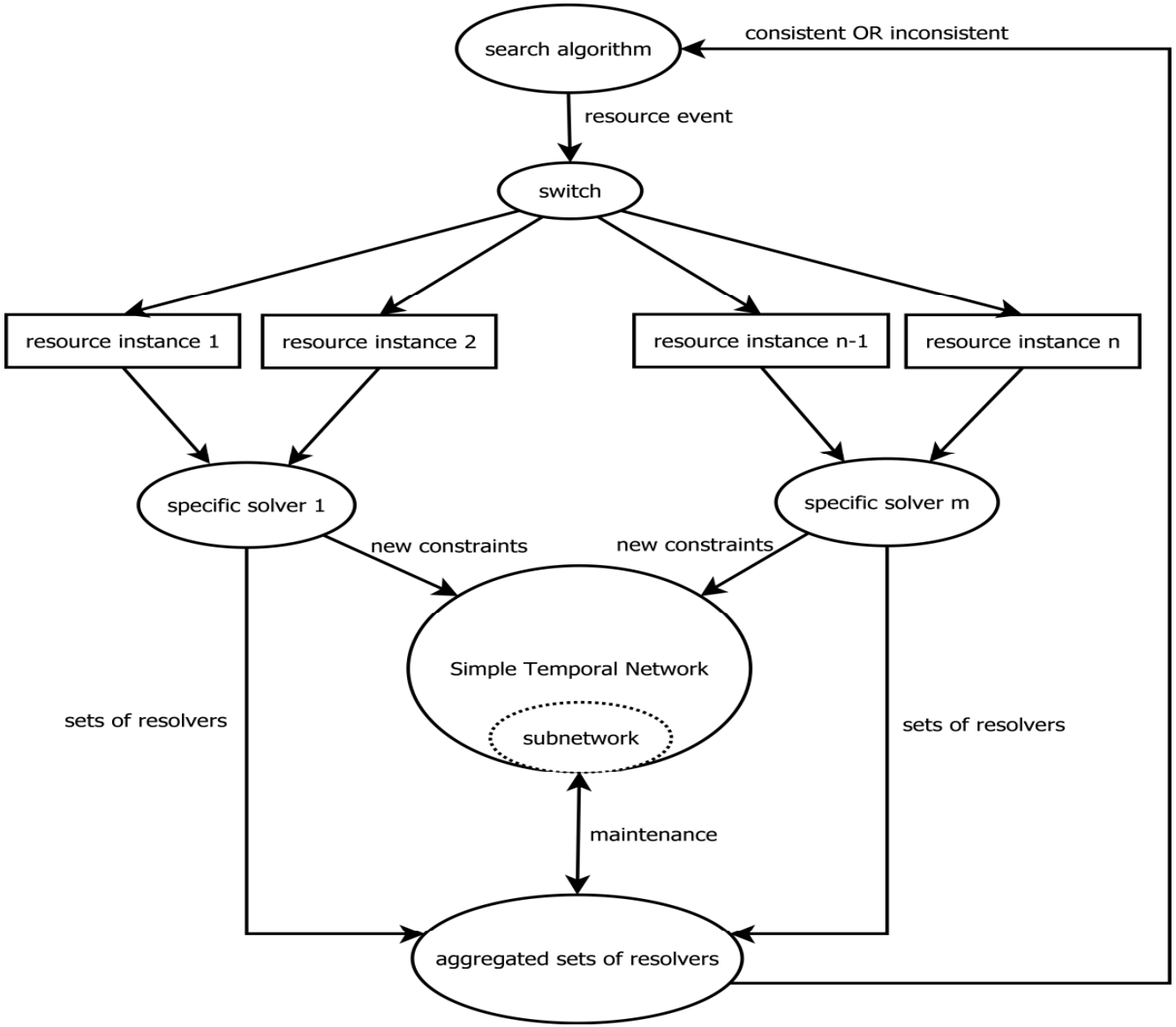


openstacks - time (s)			
	Base	SGPlan6	**Filuta[1]**
1	0.724	0.596	0.016
2	0.812	0.4	0.001
3	0.78	0.64	0.001
4	0.844	0.564	0.015
5	0.724	0.38	0.047
6	0.748	0.528	0.0061
7	0.784	0.624	0.0141
8	0.756	0.508	3.031
9	0.756	0.38	11.576
10	0.796	0.392	50.846
11	0.816	0.532	199.012
12	0.768	0.508	
13	0.848	0.432	
14	0.756	0.364	
15	0.82	0.552	
16	0.784	0.6	
17	0.856	0.4	
18	0.828	0.376	
19	0.876	0.388	
20	0.908	0.544	
21	0.916	0.524	
22	0.932	0.392	
23	0.9	0.388	
24	0.916	0.6	
25	0.904	0.412	
26	0.984	0.432	
27	0.94	0.476	
28	0.984	0.46	
29	0.988	0.528	
30	1.004	0.5	

Algoritmus


root_search
new subproblem
goal_search
searching through DTG
way_search
searching for a support
of request or change
new arc in DTG traversed trough an action
support_search
action_search
resource manager provided
action resolvers
trying action resolver
resource_search
searching for supports of
requests and changes

Resource manager



Příklad