aspeed: ASP-based Solver Scheduling

Holger Hoos Roland Kaminski Torsten Schaub
Marius Schneider

University of Potsdam

Outline

Outline

Introduction

- Goal Improving of solving performance
  via use of (multiple) competitive solvers
- Approach Algorithm schedules optimized with ASP
- Result *aspeed* allows for sequential and parallel solving
  with algorithm schedules

Outline

Toy Example

	s_1	s_2	s_3	*oracle*
i_1	1	(10)	3	1
i_2	5	(10)	2	2
i_3	8	1	(10)	1
i_4	(10)	(10)	2	2
i_5	(10)	6	(10)	6
i_6	(10)	8	(10)	8
timeouts	3	3	3	0

Timeout Minimal Schedule

	s_1	s_2	s_3	*oracle*
i_1	1	(10)	3	1
i_2	5	(10)	2	2
i_3	8	1	(10)	1
i_4	(10)	(10)	2	2
i_5	(10)	6	(10)	6
i_6	(10)	8	(10)	8
timeouts	3	3	3	0

Schedule:

- $\sigma = \{s_1 \mapsto 1, s_2 \mapsto 7, s_3 \mapsto 2\}$
- Only 1 timeout !

Timeout Minimal Schedule

	s_1	s_2	s_3	*oracle*
i_1	1	(10)	3	1
i_2	5	(10)	2	2
i_3	8	1	(10)	1
i_4	(10)	(10)	2	2
i_5	(10)	6	(10)	6
i_6	(10)	8	(10)	8
timeouts	3	3	3	0

Schedule:

- $\sigma = \{s_1 \mapsto 1, s_2 \mapsto 7, s_3 \mapsto 2\}$
- Only 1 timeout !

Timeout Minimal Schedule

	s_1	s_2	s_3	*oracle*
i_1	1	(10)	3	1
i_2	5	(10)	2	2
i_3	8	1	(10)	1
i_4	(10)	(10)	2	2
i_5	(10)	6	(10)	6
i_6	(10)	8	(10)	8
timeouts	3	3	3	0

Optimization Criterion:

$$\sigma \in \arg\max_{\sigma: S \to [0,\kappa]} |\{i \mid t(i,s) \leq \sigma(s), (i,s) \in I \times S\}|$$

$$\text{such that} \quad \sum_{s \in S} \sigma(s) \leq \kappa$$

Time Minimal Schedule

	s_1	s_2	s_3	*oracle*	*aspeed*
i_1	1	(10)	3	1	1
i_2	5	(10)	2	2	$1+2$
i_3	8	1	(10)	1	$1+2+1$
i_4	(10)	(10)	2	2	$1+2$
i_5	(10)	6	(10)	6	$1+2+6$
i_6	(10)	8	(10)	8	$(1+2+7)$
timeouts	3	3	3	0	1
average	7.3	7.5	6.2	3.3	5

Schedule:

- $\sigma = \{s_1 \mapsto 1, s_2 \mapsto 7, s_3 \mapsto 2\}$
- $\pi = \{1 \mapsto s_1, 2 \mapsto s_3, 3 \mapsto s_2\}$

Time Minimal Schedule

	s_1	s_2	s_3	*oracle*	*aspeed*
i_1	1	(10)	3	1	1
i_2	5	(10)	2	2	$1+2$
i_3	8	1	(10)	1	$1+2+1$
i_4	(10)	(10)	2	2	$1+2$
i_5	(10)	6	(10)	6	$1+2+6$
i_6	(10)	8	(10)	8	$(1+2+7)$
timeouts	3	3	3	0	1
average	7.3	7.5	6.2	3.3	5

Optimization Criterion:

$$\pi \in \arg\min_{\pi:\{1,\ldots,|S|\}\to S} \sum_{i\in I} \tau_{\sigma,\pi}(i)$$

$$\tau_{\sigma,\pi}(i) = \begin{cases} \left(\sum_{j=1}^{\min(P)-1} \sigma(\pi(j))\right) + t(i,\pi(\min(P))) & \text{if } P \neq \emptyset, \\ \kappa & \text{otherwise} \end{cases}$$

$$P = \{I \in \{1,\ldots,|S|\} \mid t(i,\pi(I)) \leq \sigma(\pi(I))\}$$

Parallel Schedules

Timeout Minimal Schedule:

$$\sigma \in \arg\max_{\sigma: S \to [0,\kappa]} |\{i \mid t(i,s) \leq \sigma(s), (i,s) \in I \times S\}|$$
$$\text{such that} \quad \sum_{s \in \eta(u)} \sigma(s) \leq \kappa \quad \text{for each } u \in U$$

Time Minimal Schedule:

$$(\pi_u)_{u \in U} \in \arg\min_{(\pi_u : \{1,\ldots,|\eta(u)|\} \to \eta(u))_{u \in U}} \sum_{i \in I} \min_{u \in U} \tau_{\sigma,\pi_u}(i)$$

$$\tau_{\sigma,\pi_u}(i) = \begin{cases} \left(\sum_{j=1}^{\min(P)-1} \sigma(\pi_u(j))\right) + t(i, \pi_u(\min(P))) & \text{if } P \neq \emptyset, \\ \kappa & \text{otherwise} \end{cases}$$

$$P = \{l \in \{1, \ldots, |\eta(u)|\} \mid t(i, \pi_u(l)) \leq \sigma(\pi_u(l))\}$$

Outline

Input

```
kappa(10).
units(2).

time(i1,s1,1).   time(i1,s2,11).  time(i1,s3,3).
time(i2,s1,5).   time(i2,s2,11).  time(i2,s3,2).
time(i3,s1,8).   time(i3,s2,1).   time(i3,s3,11).
time(i4,s1,11).  time(i4,s2,11).  time(i4,s3,2).
time(i5,s1,11).  time(i5,s2,6).   time(i5,s3,11).
time(i6,s1,11).  time(i6,s2,8).   time(i6,s3,11).
```

Timeout-Minimal Scheduling

```
solver(S)  :- time(_,S,_).
time(S,T)  :- time(_,S,T).
unit(1..N) :- units(N).
```

Timeout-Minimal Scheduling

```
solver(S)  :- time(_,S,_).
time(S,T)  :- time(_,S,T).
unit(1..N) :- units(N).

{slice(U,S,T): time(S,T): T <= K: unit(U)} 1 :- solver(S),kappa(K).
slice(S,T) :- slice(_,S,T).
```

Timeout-Minimal Scheduling

```
solver(S)  :- time(_,S,_).
time(S,T)  :- time(_,S,T).
unit(1..N) :- units(N).

{slice(U,S,T): time(S,T): T <= K: unit(U)} 1 :- solver(S),kappa(K).
slice(S,T) :- slice(_,S,T).

 :- not [ slice(U,S,T) = T ] K, kappa(K), unit(U).
```

Timeout-Minimal Scheduling

```
solver(S)  :- time(_,S,_).
time(S,T)  :- time(_,S,T).
unit(1..N) :- units(N).

{slice(U,S,T): time(S,T): T <= K: unit(U)} 1 :- solver(S),kappa(K).
slice(S,T) :- slice(_,S,T).

 :- not [ slice(U,S,T) = T ] K, kappa(K), unit(U).

solved(I,S) :- slice(S,T), time(I,S,T).
solved(I,S) :- solved(J,S), order(I,J,S).
solved(I)   :- solved(I,_).
```

Timeout-Minimal Scheduling

```
solver(S)  :- time(_,S,_).
time(S,T)  :- time(_,S,T).
unit(1..N) :- units(N).

{slice(U,S,T): time(S,T): T <= K: unit(U)} 1 :- solver(S),kappa(K).
slice(S,T) :- slice(_,S,T).

 :- not [ slice(U,S,T) = T ] K, kappa(K), unit(U).

solved(I,S) :- slice(S,T), time(I,S,T).
solved(I,S) :- solved(J,S), order(I,J,S).
solved(I)   :- solved(I,_).

#maximize { solved(I) @ 2 }.
#minimize [ slice(S,T) = T*T @ 1 ].
```

Time Minimal Scheduling

```
solver(U,S)     :- slice(U,S,_).
instance(I)     :- time(I,_,_).
unit(1..N)      :- units(N).
solvers(U,N)    :- unit(U), N := {solver(U,_)}.
solved(U,S,I)   :- time(I,S,T), slice(U,S,TS), T <= TS.
solved(U,I)     :- solved(U,_,I).
capped(U,I,S,T) :- time(I,S,T), solved(U,S,I).
capped(U,I,S,T) :- slice(U,S,T), solved(U,I), not solved(U,S,I).
capped(U,I,d,K) :- unit(U), kappa(K), instance(I), not solved(U,I).
capped(I,S,T)   :- capped(_,I,S,T).
1 { order(U,S,X) : solver(U,S) } 1 :- solvers(U,N), X = 1..N.
1 { order(U,S,X) : solvers(U,N) : X = 1..N } 1 :- solver(U,S).

solvedAt(U,I,X+1) :- solved(U,S,I), order(U,S,X).
solvedAt(U,I,X+1) :- solvedAt(U,I,X), solvers(U,N), X <= N.

mark(U,I,d,K) :- capped(U,I,d,K).
mark(U,I,S,T) :- capped(U,I,S,T), order(U,S,X), not solvedAt(U,I,X).
min(1,I,S,T)  :- mark(1,I,S,T).

less(U,I) :- unit(U), unit(U+1), instance(I),
  [min(U,I,S1,T1): capped(I,S1,T1) = T1, mark(U+1,I,S2,T2) = -T2] 0.

min(U+1,I,S,T) :- min(U,I,S,T), less(U,I).
min(U,I,S,T)   :- mark(U,I,S,T), not less(U-1,I).

#minimize [min(U,_,_,T): not unit(U+1) = T].
```

Outline

Results (in terms of timeouts)

	Crafted		*3s-Set*		*ASP-Set*	
single best	155	(51.6%)	1881	(34.4%)	28	(8.9%)
ppfolio-like	126	(42.0%)	645	(11.8%)	17	(5.4%)
satzilla	101	(34.0%)	–	(–%)	–	(–%)
aspeed (seq)	98	(32.6%)	536	(9.8%)	18	(5.7%)
aspeed (par 8)	85	(28.3%)	140	(2.5%)	8	(2.6%)

Outline

Summary

- Solving performance improved via solver schedules
- Solver schedules optimized with ASP
- *clasp* improves his own runtime !

- Sourceforge `http://potassco.sourceforge.net`
- Google+ `https://plus.google.com/102537396696345299260`

Summary

- Solving performance improved via solver schedules
- Solver schedules optimized with ASP
- *clasp* improves his own runtime !

- Sourceforge `http://potassco.sourceforge.net`
- Google+ `https://plus.google.com/102537396696345299260`

Summary

- Solving performance improved via solver schedules
- Solver schedules optimized with ASP
- *clasp* improves his own runtime !

- Sourceforge `http://potassco.sourceforge.net`
- Google+ `https://plus.google.com/102537396696345299260`