Regularization of Schedule Optimization

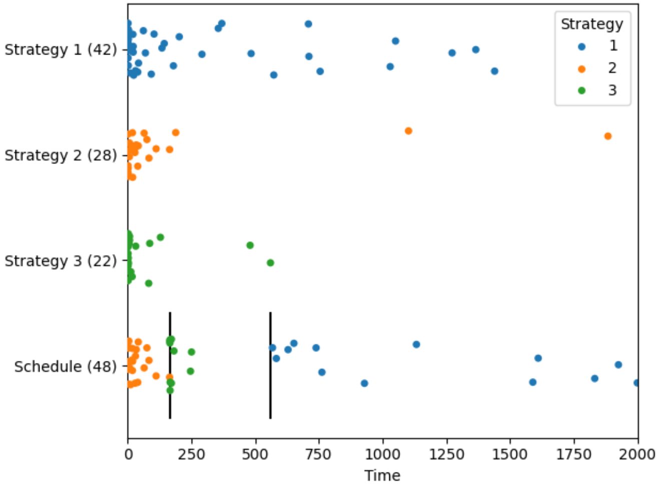
Filip Bártek, Karel Chvalovský, and Martin Suda

Czech Technical University in Prague, Czech Republic

May 22, 2024

This work was supported by the Czech Science Foundation grants 20-06390Y and 24-12759S, the European Regional Development Fund under the Czech project AI&Reasoning no. CZ.02.1.01/0.0/0.0/15_003/0000466, the project RICAIP no. 857306 under the EU-H2020 programme, and the Grant Agency of the Czech Technical University in Prague, grant no. SGS20/215/OHK3/3T/37.

Strategy schedule optimization



Strategy schedule optimization

Input

- Strategies (algorithms, configurations) S
- Problems (instances) P
- Runtime measurements $E : P \times S \to \mathbb{N} \cup \{\infty\}$
- Runtime budget $T \in \mathbb{N}$

Output

Schedule $\mathfrak{s} : S \to \mathbb{N}$ such that $\sum_{s \in S} \mathfrak{s}(s) \leq T$

Optimization criterion

Maximize the number of solved problems:

$$|\{p \in P \mid \exists s \in S : E(p, s) \leq \mathfrak{s}(s)\}|$$

Our dataset: Vampire + TPTP

- Target solver: automatic theorem prover Vampire
- 1096 strategies (configurations of Vampire)
- 7866 first-order logic (FOL) problems from TPTP
- $8\,621\,136 = 1096 \cdot 7866$ solver runs
  - Time limit: 2000 to 256 000 CPU megainstructions (Mi)

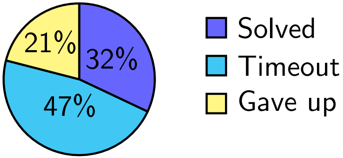


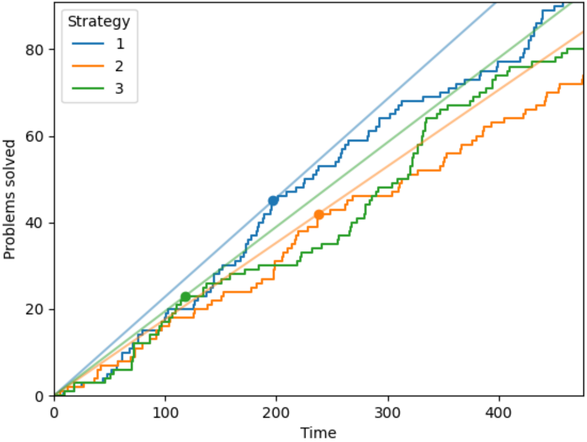
Figure: Distribution of run statuses

Public dataset: `https://zenodo.org/records/10814478`

Our schedule optimization scenario

- Training problems: $|P_{train}| \approx 80\,\% \cdot |P| \approx 6293$
- Strategies: $|S| \approx 829$

Greedy schedule optimization



Regularization methods

Regularization method	Parameter	Default
Additive slack	$b \in \mathbb{N}$	$b = 0$
Multiplicative slack	$w \geq 1$	$w = 1$
Temporal reward adjustment	$0 \leq \alpha$ (reward exponent)	$\alpha = 1$
Diminishing problem rewards	$0 \leq \beta \leq 1$ (discount factor)	$\beta = 0$

Slice extension with reward adjustment

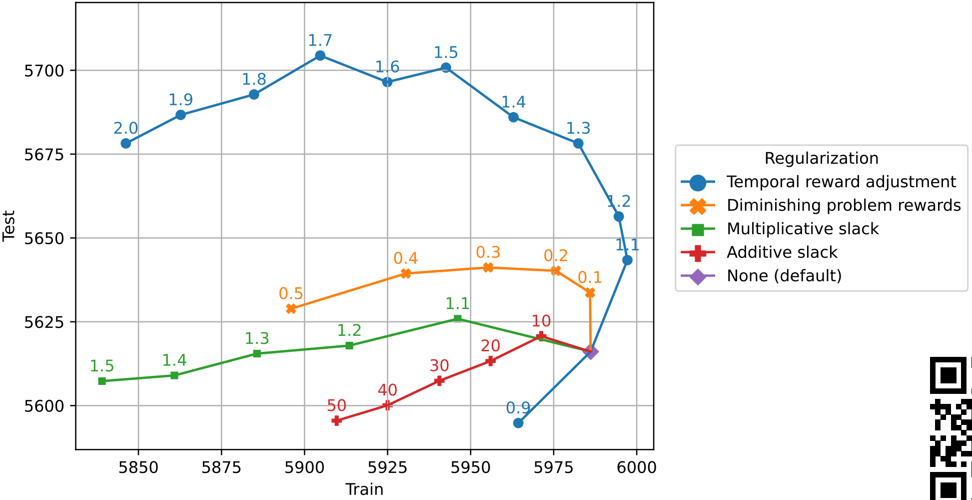
$$s, t \leftarrow \operatorname{argmax}_{s \in S, 0 < t \leq T'} \frac{\left(\sum_{p \in P} [\![\mathfrak{s}(s) < E(p,s) \leq \mathfrak{s}(s) + t]\!] \beta^{\#covered(p)}\right)^{\alpha}}{t}$$

Schedule post-processing with slack

for all $s \in S$ such that $\mathfrak{s}(s) > 0$ **do**

$\quad \mathfrak{s}(s) \leftarrow \mathfrak{s}(s) \cdot w + b$

Regularization for $T = 64\,000\,\mathrm{Mi}$



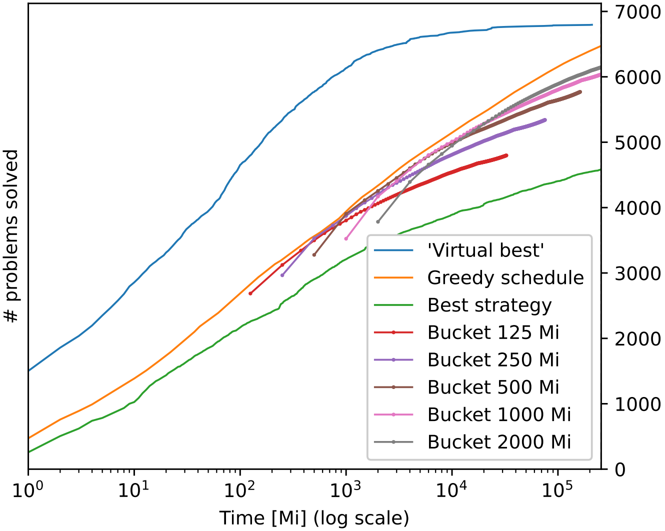
Paper (IJCAR 2024): `https://arxiv.org/abs/2403.12869`

Greedy schedule optimization

Input: Problems P, strategies S, runtimes E, budget T
Output: Schedule $\mathfrak{s} : S \to \mathbb{N}$

$\forall s \in S : \mathfrak{s}(s) \leftarrow 0$ ▷ Start with an empty schedule
$P' \leftarrow P$ ▷ Remaining problems
$T' \leftarrow T$ ▷ Remaining budget
repeat
  $s, t \leftarrow \operatorname{argmax}_{s \in S, 0 < t \leq T'} \frac{|\{p \in P' | E(p,s) \leq \mathfrak{s}(s) + t\}|}{t}$ ▷ Maximize new problems per time
  $\mathfrak{s}(s) \leftarrow \mathfrak{s}(s) + t$ ▷ Extend the schedule
  $P' \leftarrow \{p \in P' | E(p, s) > \mathfrak{s}(s)\}$ ▷ Remove the solved problems
  $T' \leftarrow T' - t$
until $T' = 0$ or $P' = \emptyset$
return $\mathfrak{s}$

VBS



7000
6000
5000
4000
3000
2000
1000
0
problems solved
10^0
10^1
10^2
10^3
10^4
10^5
Time [Mi] (log scale)
'Virtual best'
Greedy schedule
Best strategy
Bucket 125 Mi
Bucket 250 Mi
Bucket 500 Mi
Bucket 1000 Mi
Bucket 2000 Mi

Perfect schedule optimization

NP-hard

Integer programming

Maximize $\sum_{p \in P} \text{Solved}(p)$ subject to:

- $\forall p \in P : \text{Solved}(p) \to \bigvee_{s \in S} \text{SolvedBy}(p, s)$
- $\forall p \in P, \forall s \in S : \text{SolvedBy}(p, s) \to E(p, s) \leq \mathfrak{s}(s)$
- $\sum_{s \in S} \mathfrak{s}(s) \leq T$

Output schedule: $\mathfrak{s} : S \to \mathbb{N}$

Strategy collection

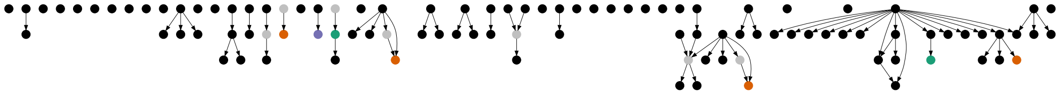
Repeat:

1. Sample strategy s
2. Determine time limit t
3. Sample problem p
4. Attempt to solve p with s in time limit t. If success:
    - Optimize s on p by local search to reduce solving time
    - Evaluate s on all problems with time limit $2t$
    - Store s and the evaluation results

Strategy sampling

103 strategy parameters

- Vampire: 96
  - • Categorical: 89
  - Numeric: 7
    - • Ratio: 4
    - • Floating point uniform: 2
    - • Integer uniform: 1
- • Auxiliary (categorical): 7



Parameter distributions: Biased in favor of strong strategies

Time limit

- Time unit: 10^6 CPU instructions (megainstruction, Mi)
  - 1 second is approximately 2000 Mi on our hardware
- Deterministic sampling: $1000 \times$ Luby sequence (1, 1, 2, 1, 1, 2, 4, ...)
- Initial time limits: 1000, 1000, 2000, 1000, 1000, 2000, 4000, 1000, 1000, 2000, 1000, 1000, 2000, 4000, 8000, ...

Problem sampling

TPTP FOF theorems and unknown – 7866 problems

- Status
  - Theorem (THM, CAX, UNS): 7711
  - Unknown (UNK, OPN): 154
- With equality: 1276

Distribution: Biased in favor of low TPTP rating

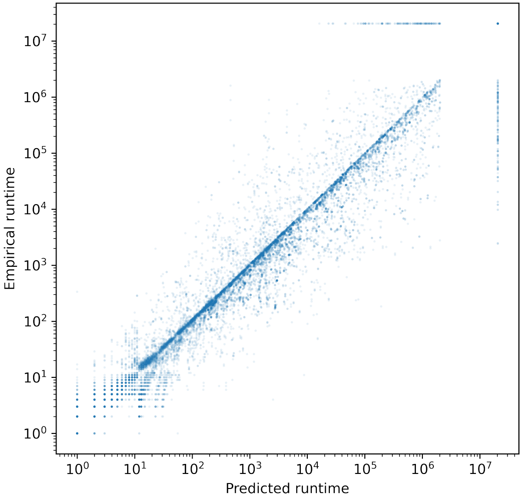
Simulated vs. empirical success

How well does our estimated schedule performance model the actual performance?

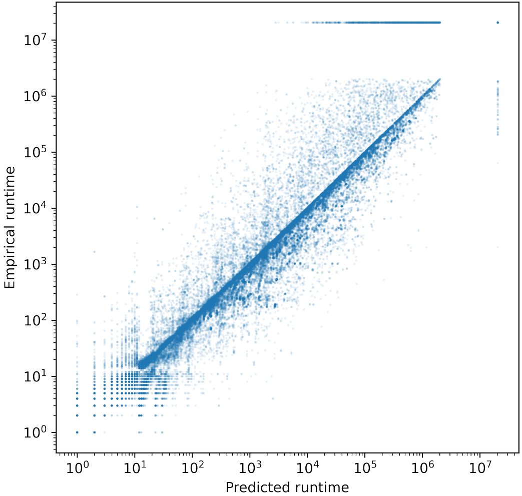
Table: Simulated vs. empirical success (total point evaluations: 23598)

Empirical	Predicted	
	Success	Timeout
Success	18 871	224
Timeout	212	4291

Simulated vs. empirical runtime on the test problems



Simulated vs. empirical runtime on the train problems



Diminishing problem rewards

Input: Problems P, strategies S, runtimes E, budget T, discount factor β $(0 \le \beta \le 1)$
Output: Schedule $\mathfrak{s} : S \to \mathbb{N}$

```
∀s ∈ S : 𝔰(s) ← 0                                   ▷ Start with an empty schedule
∀p ∈ P : k(p) ← 0                                   ▷ Number of times the problem has been covered
T′ ← T                                              ▷ Remaining budget
repeat
    s, t ← argmax_{s∈S, 0<t≤T′} (Σ_{p∈P} [[𝔰(s) < E(p,s) ≤ 𝔰(s)+t]] β^{k(p)}) / t     ▷ Reward per time
    for all p ∈ P such that 𝔰(s) < E(p, s) ≤ 𝔰(s) + t do
        k(p) ← k(p) + 1
    𝔰(s) ← 𝔰(s) + t
    T′ ← T′ − t
until no more problems can be covered
return 𝔰
```

$$s, t \leftarrow \operatorname{argmax}_{s \in S, 0 < t \le T'} \frac{\sum_{p \in P} [\![\mathfrak{s}(s) < E(p,s) \le \mathfrak{s}(s) + t]\!] \beta^{k(p)}}{t}$$

Slice extension trick

Input: Problems P, strategies S, runtimes E, budget T
Output: Schedule $\mathfrak{s} : S \to \mathbb{N}$

```
∀s ∈ S : 𝔰(s) ← 0                                          ▷ Start with an empty schedule
P′ ← P                                                      ▷ Remaining problems
T′ ← T                                                      ▷ Remaining budget
repeat
    s, t ← argmax_{s∈S, 0<t≤T′} |{p∈P′ | E(p,s) ≤ t}| / t   ▷ Maximize new problems per time
    𝔰(s) ← t                                                ▷ Extend the schedule
    P′ ← {p ∈ P′ | E(p,s) ≥ 𝔰(s)}                           ▷ Remove the solved problems
    T′ ← T′ − t
until no new problems can be solved
return 𝔰
```

$$s, t \leftarrow \operatorname{argmax}_{s \in S, 0 < t \leq T'} \frac{|\{p \in P' | E(p,s) \leq t\}|}{t}$$

Slice extension trick

Input: Problems P, strategies S, runtimes E, budget T
Output: Schedule $\mathfrak{s} : S \to \mathbb{N}$

$\forall s \in S : \mathfrak{s}(s) \leftarrow 0$ ▷ Start with an empty schedule
$P' \leftarrow P$ ▷ Remaining problems
$T' \leftarrow T$ ▷ Remaining budget
repeat
  $s, t \leftarrow \operatorname{argmax}_{s \in S, 0 < t \leq T'} \frac{|\{p \in P' | E(p,s) \leq \mathfrak{s}(s) + t\}|}{t}$ ▷ Maximize new problems per time
  $\mathfrak{s}(s) \leftarrow \mathfrak{s}(s) + t$ ▷ Extend the schedule
  $P' \leftarrow \{p \in P' | E(p, s) \geq \mathfrak{s}(s)\}$ ▷ Remove the solved problems
  $T' \leftarrow T' - t$
until no new problems can be solved
return $\mathfrak{s}$

Our dataset: Vampire + TPTP

- Target solver: automatic theorem prover Vampire
  - Strategy space dimension: 103 parameters
- 1096 strategies (configurations of Vampire)
- 7866 FOL problems from TPTP
  - Solved by some strategy: 6796 in 256 000 Mi, 6405 in 2000 Mi
  - Solved by the best strategy: 4582
  - Solved by the default strategy: 4264
- $8\,621\,136 = 1096 \cdot 7866$ solver runs
  - Time limit: 2000 to 256 000 CPU megainstructions (Mi)
    - Approximately 1 to 128 wallclock seconds