PaSAT – Parallel SAT-Checking with Lemma Exchange

Carsten Sinz, Wolfgang Blochinger and Wolfgang Küchlin
Symbolic Computation Group
University of Tübingen, Germany

http://www-sr.informatik.uni-tuebingen.de

Introduction

Goal:

Speed up DP algorithm for encodings of real-world combinatorial problems

Instruments:

- ★ Parallelization (dyn. search-space partitioning)
- ★ Learning (lemma generation)

PaSAT:

Combination of both acceleration methods

Basic Davis-Putnam Algorithm

```
boolean DP(Clause Set S)
{   while (S contains a unit clause {L}) {    // unit prop.
        delete from S all clauses containing L;    // u. subs.
        delete ¬L from all clauses in S;           // u. res.
    }
    if (∅ ∈ S) return false;
    if (S = ∅) return true;
    choose a literal L occurring in S;
    if (DP(S ∪ {{L}}) return true;
    else return DP(S ∪ {{¬L}});
}
```

Recent Improvements on DP-based Solvers

- Efficient implementation of UP and backtracking
  - Head/tail lists [H.Zhang, Stickel ´96]
  - Watched literals [Moskewicz *et al*. ´01]
- Improved variable selection strategies
  - Look-ahead [Li ´97], “decaying sum” [Moskewicz *et al*. ‘01], combinations [H.Zhang ´97]
- Randomization, search restarts
  - Overcome limitations of heuristics (influenced by local search algorithms) [Gomes/Selman/Kautz ´98]
- Clause set „compression“
  - Deletion of subsumed clauses [L.Zhang, Malik ´01]
- Lemma generation [Marques-Silva, Sakallah ´96]

Parallelization Method

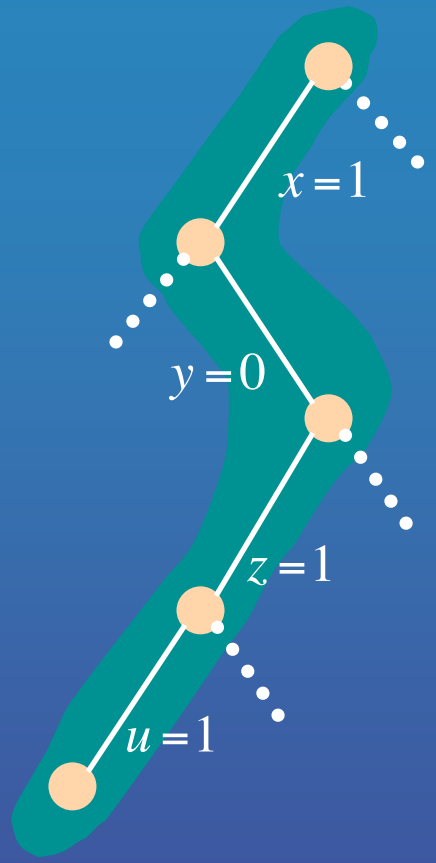
- *Guiding path* [H. Zhang *et al.*, ´96] describes state of search, e.g.

$$((x,\mathrm{B}),(\bar{y},\mathrm{N}),(z,\mathrm{B}),(u,\mathrm{B}))$$

- Partitioning of search-space at each (_,B) entry possible, e.g.

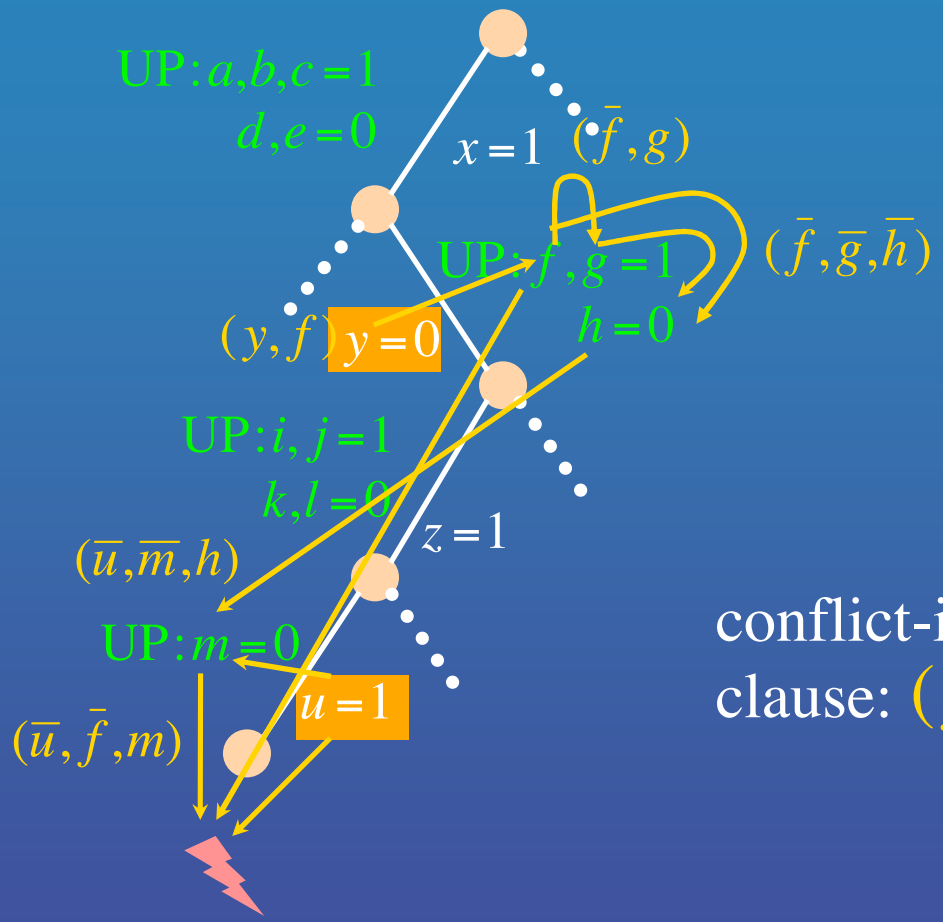
$$((x,\mathrm{N}),(\bar{y},\mathrm{N}),(z,\mathrm{B}),(u,\mathrm{B}))$$

$$((\bar{x},\mathrm{N}))$$



Lemma Generation [Marques-Silva, Sakallah]

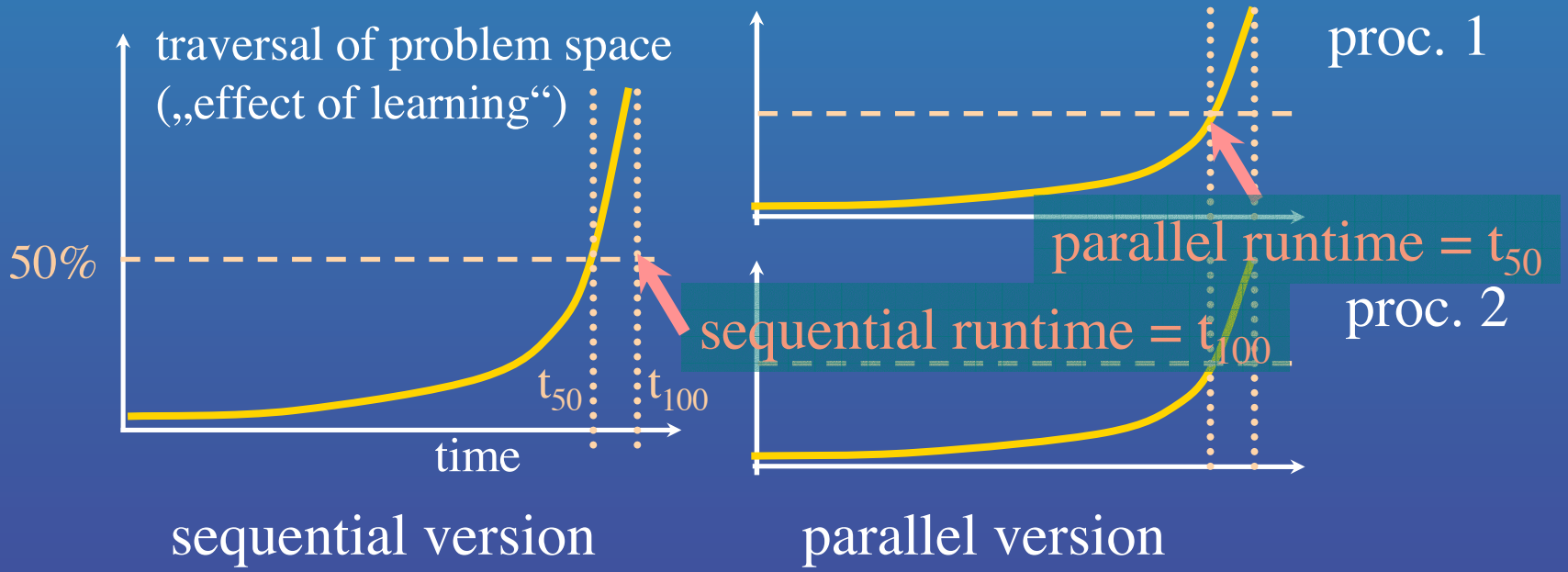
$(\bar{u}, \bar{f}, m)$
$(\bar{u}, \bar{m}, h)$
$(\bar{f}, \bar{g}, \bar{h})$
(y, f)
$(\bar{f}, g)$
$\vdots$



conflict-induced clause: $(y, \bar{u})$

Challenges of Combining Both Ideas

- Cooperation needed to make full use of lemmas
- Acceleration by lemma generation may limit speed-ups attainable by parallelization:



Analyzing the Learning Effect

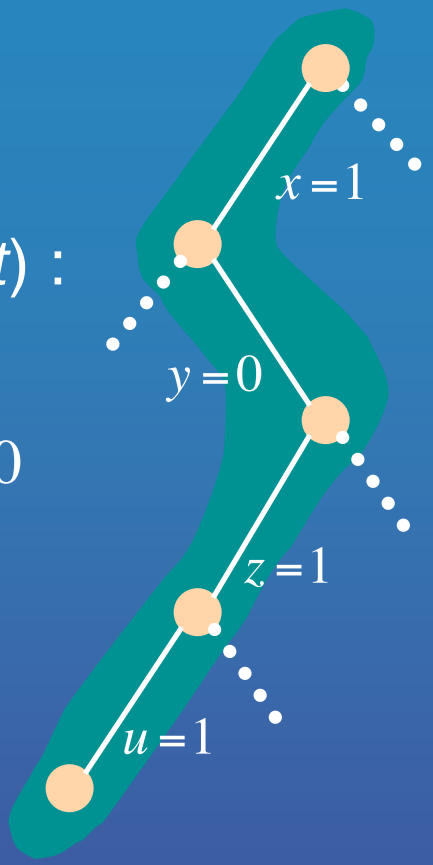
- Fraction f(t) of traversed search space at time *t* described by current guiding path *P*(*t*) :

$$P(t) = ((L_1, b_1), \ldots, (L_d, b_d))$$

$$f(t) = \sum_{i=1}^{d} w(b_i) \cdot 2^{-i} \text{ where } w(\mathrm{N}) = 1, w(\mathrm{B}) = 0$$

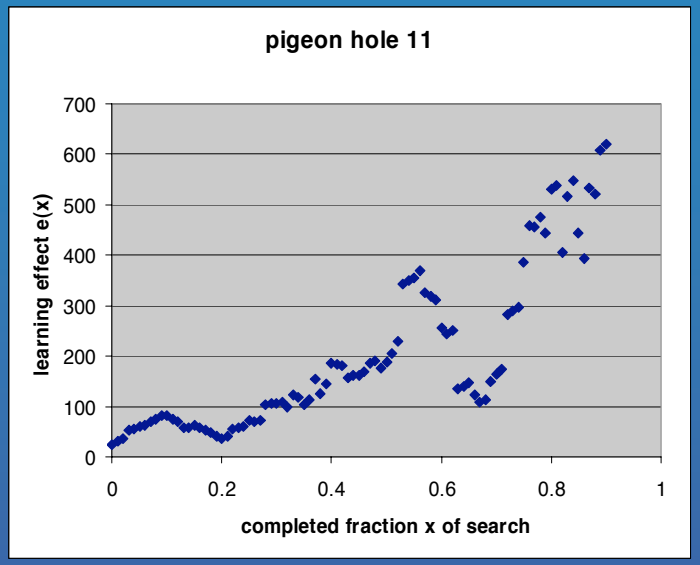
- Effect e(x) of learning when fraction *x* of search is completed:

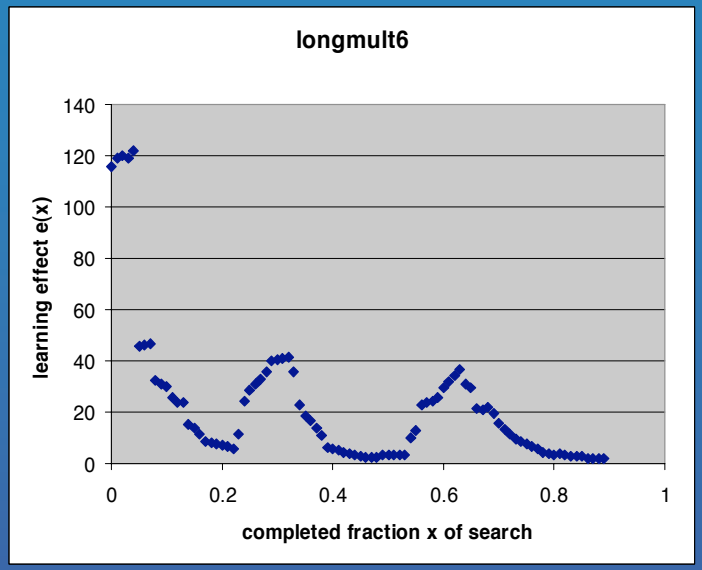
$$e(x) = \frac{\frac{df_l}{dt} \circ f_l^-}{\frac{df_n}{dt} \circ f_n^-}(x), \quad x \in [0,1]$$



$$((x, \mathrm{B}), (\bar{y}, \mathrm{N}), (z, \mathrm{B}), (u, \mathrm{B}))$$

Analyzing the Learning Effect (cont´d)





$t_{\text{seq},n} = 3606.13$ $t_{\text{seq},l} = 41.10$ $t_{\text{seq},n} = 27.31$ $t_{\text{seq},l} = 32.32$

$t_{\text{par},n} = 996.47$ $t_{\text{par},l} = 55.02$ $t_{\text{par},n} = 6.85$ $t_{\text{par},l} = 4.66$

$s_{\text{par/seq},n} = 3.62$ $s_{\text{par/seq},l} = 0.75$ $s_{\text{par/seq},n} = 3.99$ $s_{\text{par/seq},l} = 6.94$

(parallel times on 4 processors)

Parallelization Platform

DOTS (Distributed Object-Oriented Threads System):

- Fork/join-(threads-)paradigm for both parallel and distributed computing
- C++
- Supported Systems:
  - Windows 98/NT/2000
  - Solaris, IRIX, AIX, FreeBSD, Linux
  - IBM zSeries z/OS
- Different load-distribution schemes, e.g.
  - work-sharing
  - work-stealing

Experimental Results

problem instance	t_{Seq}	t_{Par}	t_{PaSAT}	$s_{Par/Seq}$	$s_{Par/PaSAT}$
longmult7	73.78	22.24	15.49	3.32	4.76
longmult8	176.91	51.54	42.76	3.43	4.14
longmult9	291.00	79.51	77.78	3.66	3.74
longmult10	414.49	113.92	138.71	3.64	2.99
longmult11	541.14	145.81	179.28	3.71	3.02
longmult12	646.43	163.66	162.78	3.95	3.97
longmult13	809.67	202.92	227.93	3.99	3.55
longmult14	929.21	242.13	248.13	3.84	3.74

Sun ES450, 4 UltraSparcII processors @ 400MHz, 1GB

Conclusion

- Combination of learning/parallelization:
  - Cooperation by exchanging suitable lemmas
  - Makes use of networks of multiprocessor workstations
  - Delivers good results on real-world SAT-instances
- Preliminary analysis of learning behavior:
  - Allows graphical presentation of learning effect over time
- Main result:

Parallelization and learning
perfectly complement each other.