Evaluating CDCL Restart Schemes

Armin Biere   Andreas Fröhlich

Johannes Kepler University, Linz, Austria

Pragmatics of SAT 2015

POS'15

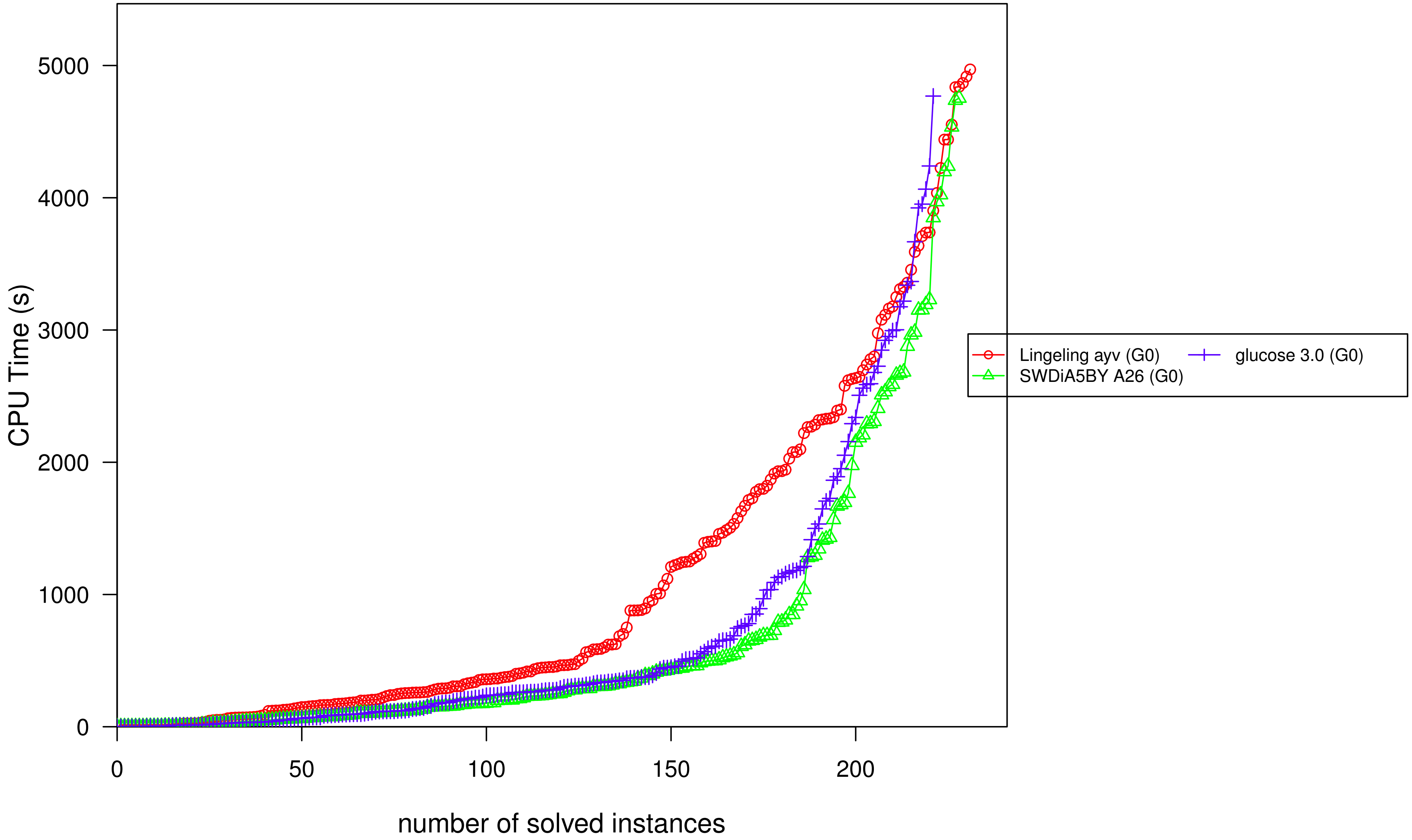
The University of Texas at Austin

Austin, TX, USA

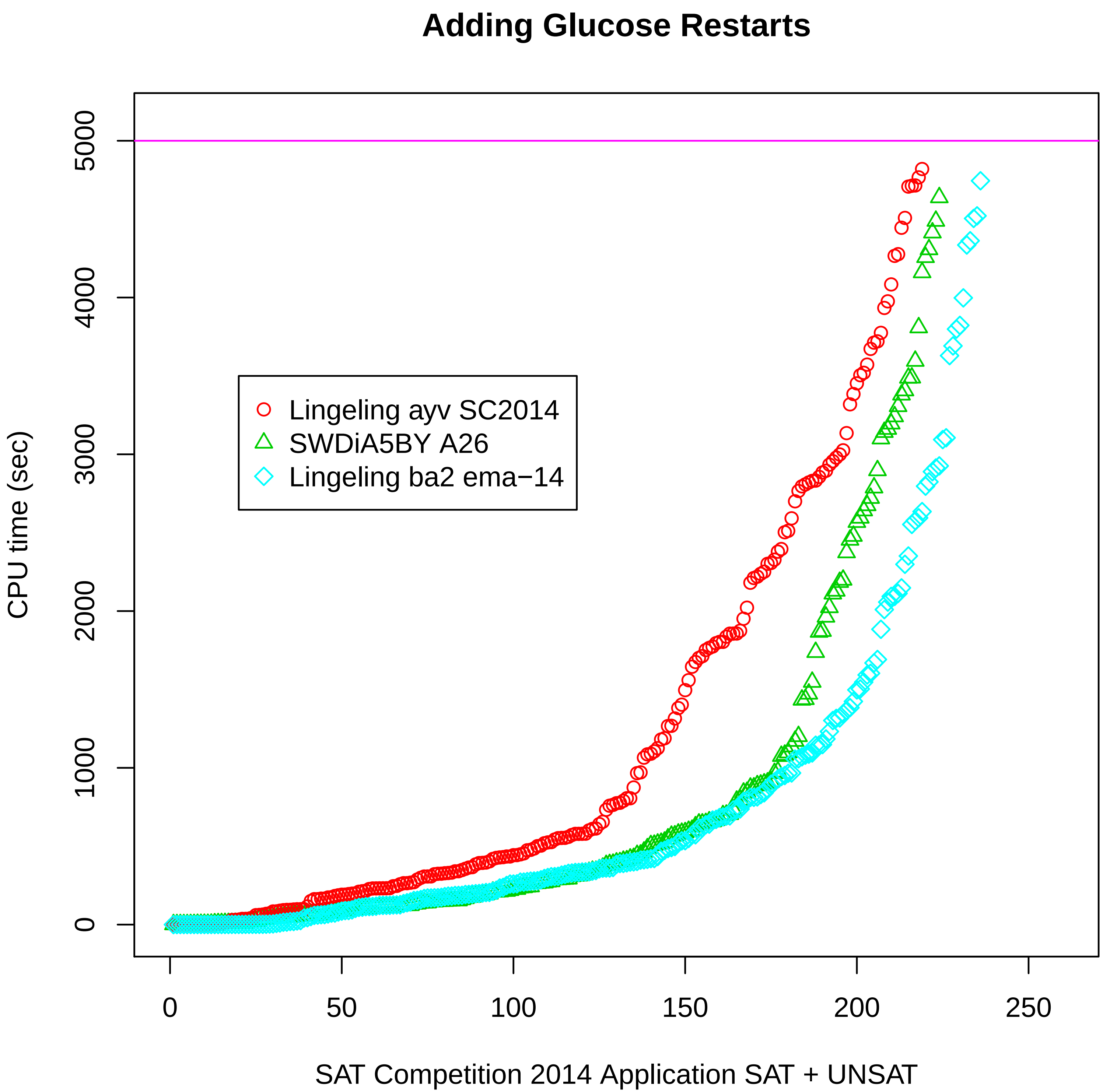
Wednesday, 23rd September, 2015

Original SAT Competition 2014 Application SAT+UNSAT Cactus

Number of solved instances within a given amount of CPU time



http://satcompetition.org/edacc/sc14



Adding Glucose Restarts
Lingeling ayv SC2014
SWDiA5BY A26
Lingeling ba2 ema−14
CPU time (sec)
0
1000
2000
3000
4000
5000
0
50
100
150
200
250
SAT Competition 2014 Application SAT + UNSAT

- Lingeling actually barely won
  - only for long time limit of 5000 seconds
  - for 900 seconds: no chance
- two main reasons
  - selected benchmark biased towards decendants of Glucose / MiniSAT
  - but Glucose restarts are important for many (selected) benchmarks
- this paper is about lessons learned while
  - porting the Glucose restart scheme to Lingeling
  - and **simplifying** by

    using *exponential moving averages* (*EMA*) application track instances clustered in buckets (by the organizers):

2d-strip-packing (4), argumentation (20), bio (11),

crypto-aes (8), crypto-des (7), crypto-gos (9),
crypto-md5 (21), crypto-sha (29), crypto-vpmc (4),

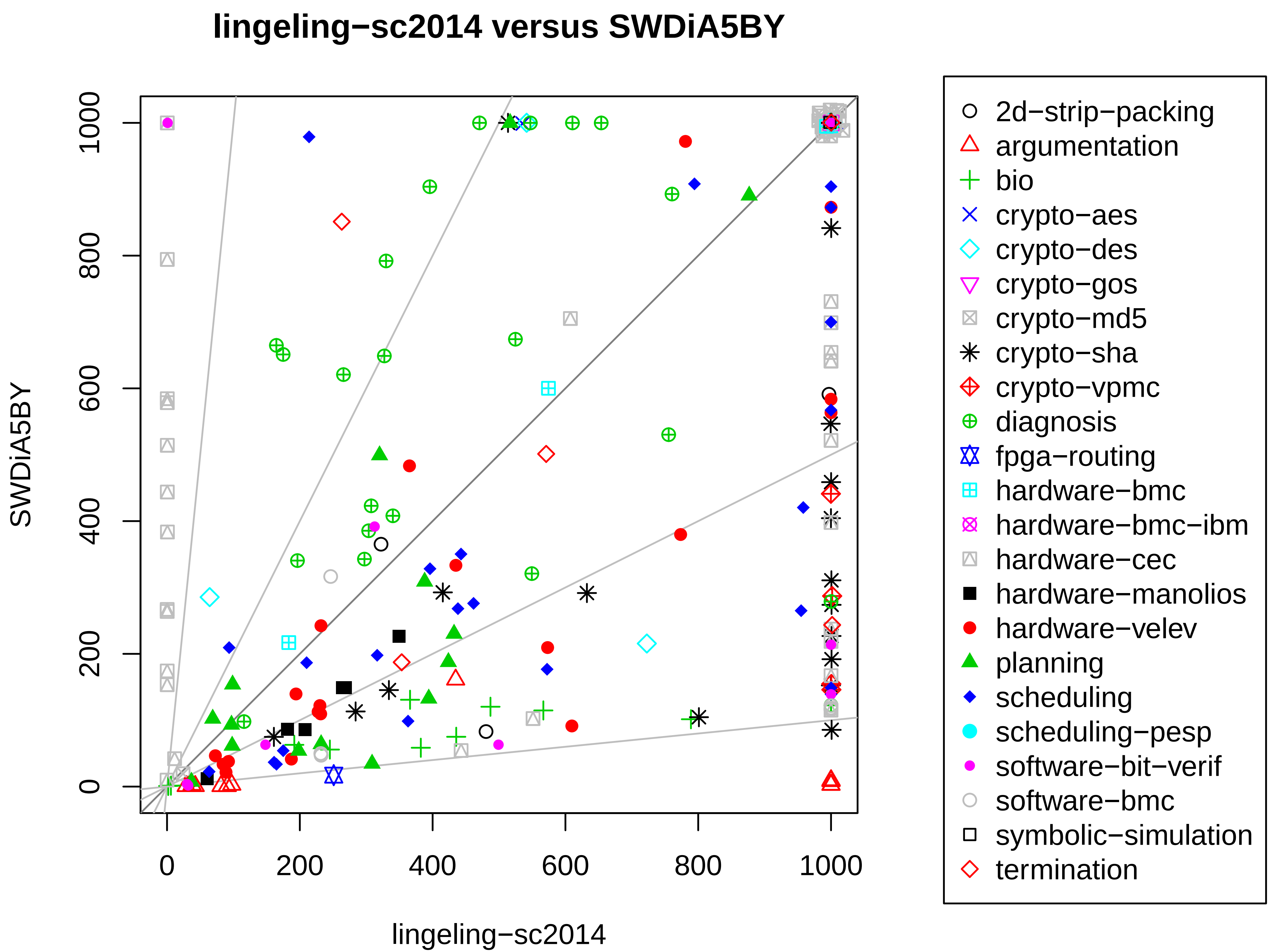
diagnosis (28), fpga-routing (1),

hardware-bmc (4), hardware-bmc-ibm (18), hardware-cec (30),
hardware-manolios (6), hardware-velev (27),

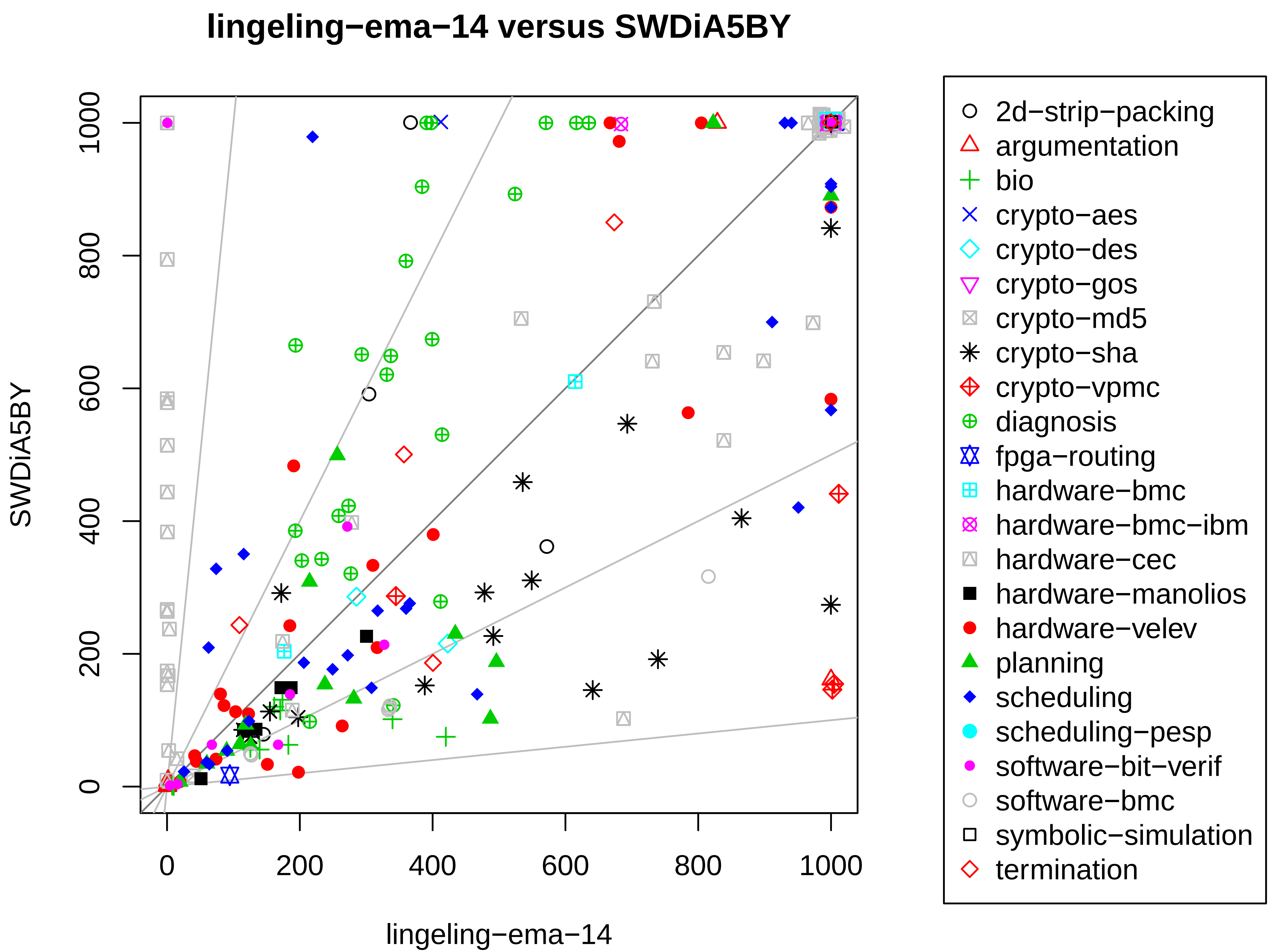
planning (19), scheduling (30), scheduling-pesp (3),

software-bit-verif (9), software-bmc (6), symbolic-simulation (1), termination (5)

in total **300** instances clustered in **23** buckets



lingeling−sc2014 versus SWDiA5BY
SWDiA5BY
lingeling−sc2014
0
200
400
600
800
1000
2d−strip−packing
argumentation
bio
crypto−aes
crypto−des
crypto−gos
crypto−md5
crypto−sha
crypto−vpmc
diagnosis
fpga−routing
hardware−bmc
hardware−bmc−ibm
hardware−cec
hardware−manolios
hardware−velev
planning
scheduling
scheduling−pesp
software−bit−verif
software−bmc
symbolic−simulation
termination



lingeling−ema−14 versus SWDiA5BY
SWDiA5BY
lingeling−ema−14
0
200
400
600
800
1000
2d−strip−packing
argumentation
bio
crypto−aes
crypto−des
crypto−gos
crypto−md5
crypto−sha
crypto−vpmc
diagnosis
fpga−routing
hardware−bmc
hardware−bmc−ibm
hardware−cec
hardware−manolios
hardware−velev
planning
scheduling
scheduling−pesp
software−bit−verif
software−bmc
symbolic−simulation
termination

Restarts in CDCL


```
Status run_CDCL_loop_with_restarts () {
  for (;;) {
    if (bcp ()) {
          if (restarting ()) restart ();
      else if (!decide ()) return SATISFIABLE;
    } else {
      conflicts++;
      if (!analyze ()) return UNSATISFIABLE;
    }
  }
}
```

- run BCP and conflict analysis (including learning) until completion
- restart if restart policy implemented in `restarting` says so
- usually based on a global `conflicts` counter
- otherwise pick next decision (unless all are assigned)

Restart Scheme Examples


```
bool restarting () {
  return conflicts >= limit;
}

void static_uniform_restart () {
  restarts++;
  limit = conflicts + interval;
  backtrack (0);
}

void static_geometric_restart () {
  limit = conflicts + interval * pow (1.5, ++restarts);
  backtrack (0);
}

void luby_restart () {
  limit = conflicts + interval * luby (++restarts);
  backtrack (0);
}
```

Restart Scheme Classification

- **static schemes**
  - fixed schedule of restarts only based on conflicts counter
    - **uniform intervals**: wait a fixed number of conflicts after each restart
    - **non-uniform restart intervals**
      - number of performed restarts determines next interval (in terms of conflicts)
      - arithmetically or geometrically increasing actual interval
      - Luby scheme (also known as reluctant doubling)
      - inner-outer scheme
- **dynamic schemes**
  - *agility* based restart blocking
  - *local restarts* (not discussed in the paper nor the talk)
  - *reusing the trail* implicitly also blocks restarts (even partially)
  - *Glucose restart* scheme (focus here)

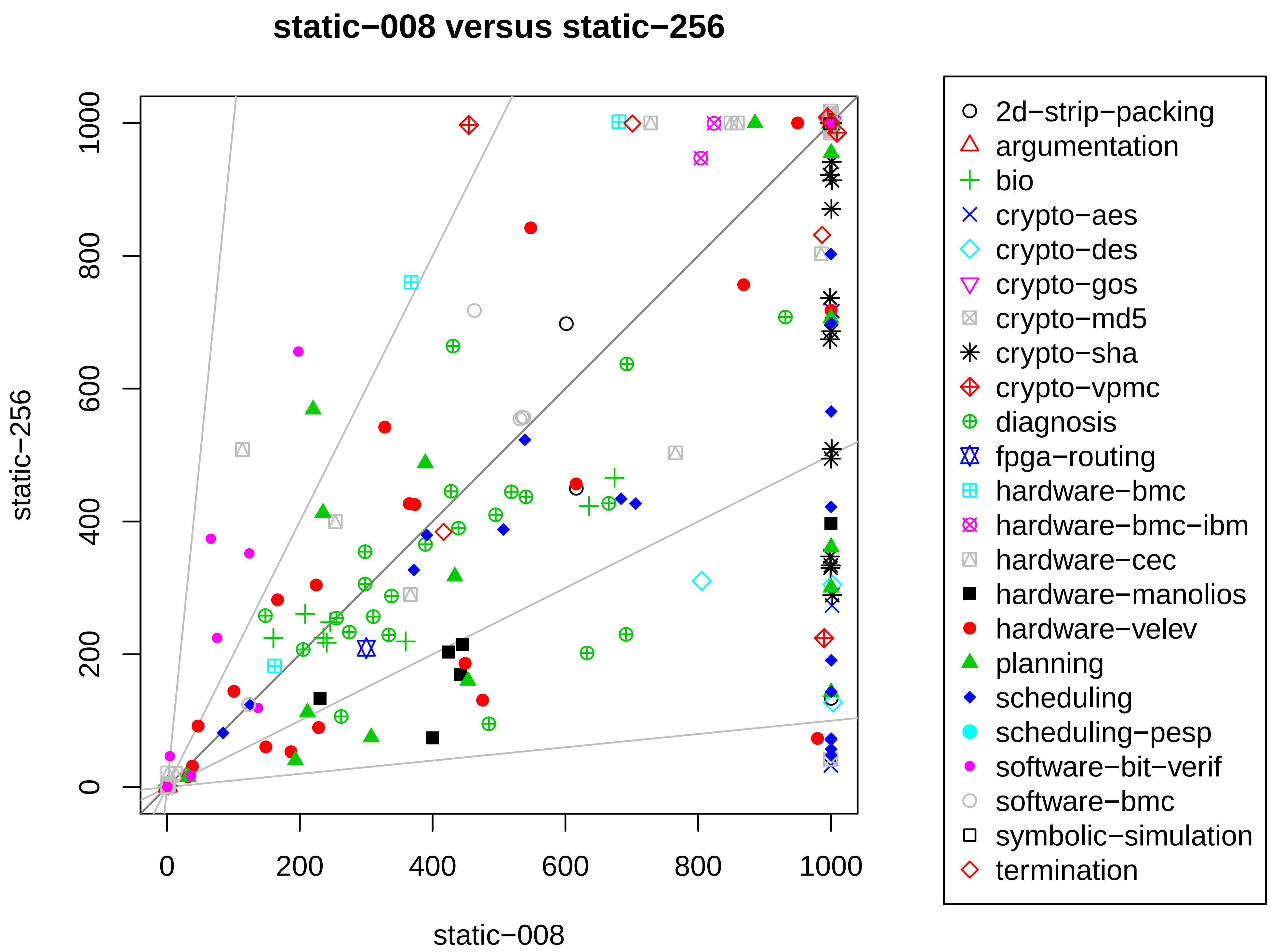
Comparing Static Uniform Restart Schemes

r	002	004	008	016	032	064	128	256	512
tot	122	127	139	144	144	161	163	**168**	158
sat	40	41	49	56	56	73	76	**83**	79
uns	82	86	**90**	88	88	88	87	85	79

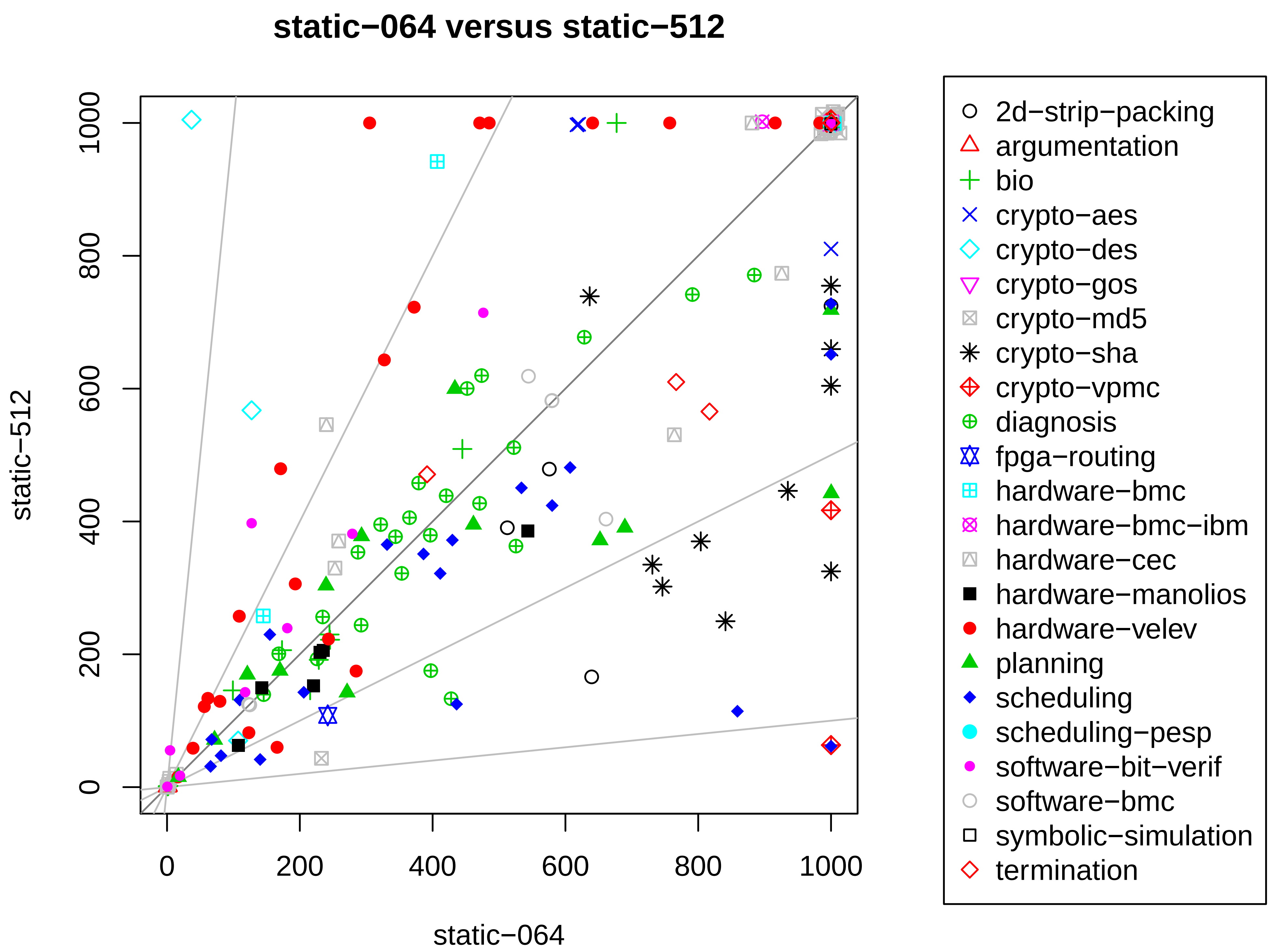
r	002	004	008	016	032	064	128	256	512
2d-strip-packing	0/2	0/2	0/2	0/2	0/2	1/2	1/2	1/2	**2/2**
crypto-sha	0/0	0/0	0/0	0/0	1/0	7/0	11/0	**13/0**	10/0
hardware-cec	0/22	0/23	**0/24**	0/22	0/22	0/23	0/22	0/21	0/21
hardware-manolios	0/4	0/5	0/5	0/5	**0/6**	**0/6**	**0/6**	**0/6**	**0/6**
hardware-velev	5/9	6/10	8/11	8/12	8/12	**8/13**	8/12	8/11	8/6
planning	6/3	6/5	7/4	7/4	8/4	9/3	9/4	**11/4**	10/4
scheduling	1/7	0/7	1/7	4/7	6/7	9/7	9/7	11/7	**12/7**

SAT / UNSAT

underlined best



static−008 versus static−256
static−256
static−008
0
200
400
600
800
1000
0
200
400
600
800
1000
2d−strip−packing
argumentation
bio
crypto−aes
crypto−des
crypto−gos
crypto−md5
crypto−sha
crypto−vpmc
diagnosis
fpga−routing
hardware−bmc
hardware−bmc−ibm
hardware−cec
hardware−manolios
hardware−velev
planning
scheduling
scheduling−pesp
software−bit−verif
software−bmc
symbolic−simulation
termination

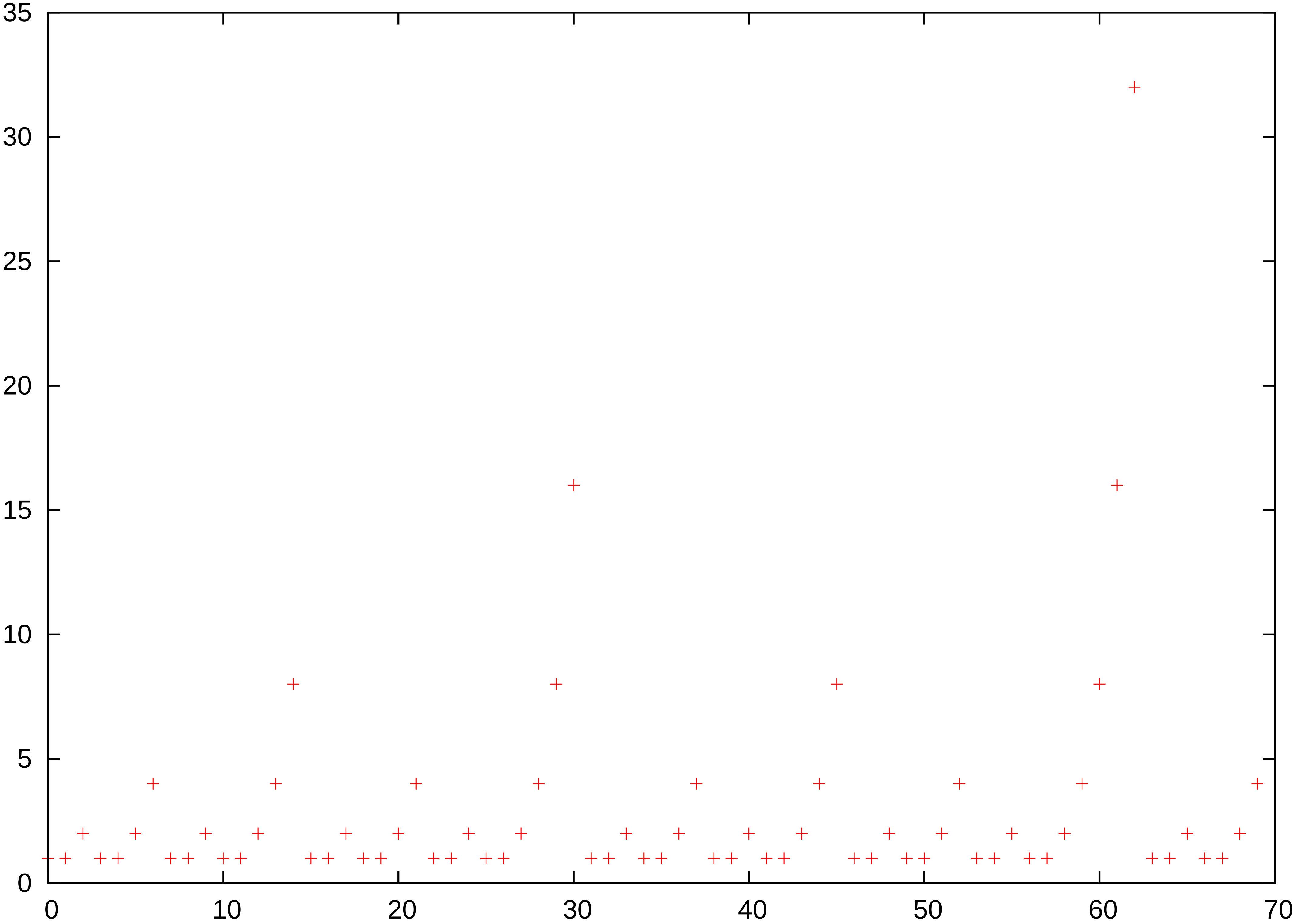


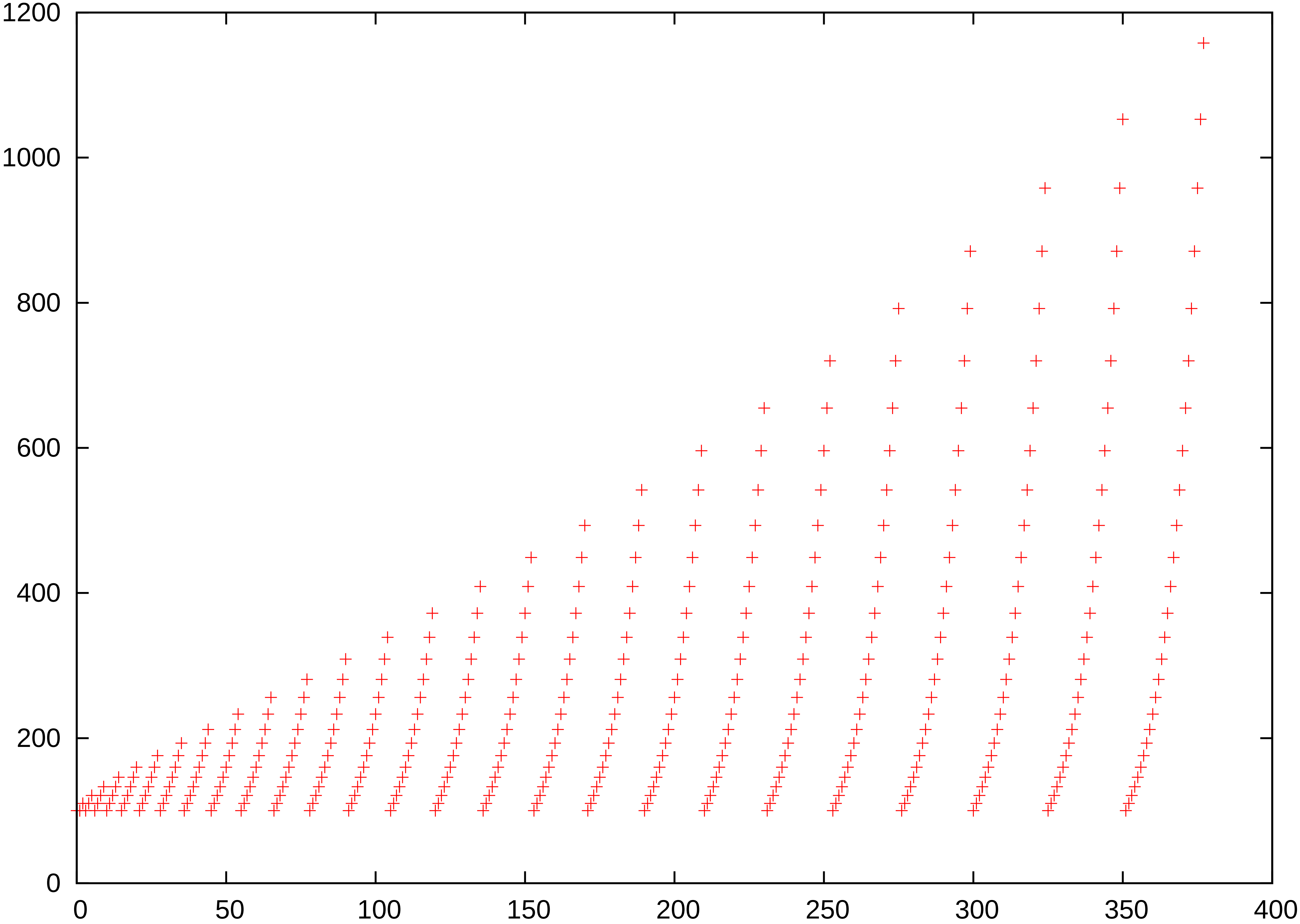
static−064 versus static−512
static−512
static−064
0
200
400
600
800
1000
0
200
400
600
800
1000
2d−strip−packing
argumentation
bio
crypto−aes
crypto−des
crypto−gos
crypto−md5
crypto−sha
crypto−vpmc
diagnosis
fpga−routing
hardware−bmc
hardware−bmc−ibm
hardware−cec
hardware−manolios
hardware−velev
planning
scheduling
scheduling−pesp
software−bit−verif
software−bmc
symbolic−simulation
termination

Comparing Static but Non-Uniform Restart Schemes

	luby-b						inner-outer					
	01	02	04	08	16	32	001	002	004	008	032	128
tot	156	**168**	161	163	160	159	**161**	**161**	158	153	154	150
sat	72	**80**	74	77	74	74	80	**81**	77	74	76	76
uns	84	**88**	87	86	86	85	**81**	80	81	79	78	74
avgc	9	17	31	58	108	203	443	509	601	732	1084	1740

avgc = average restart interval (over all instances) in conflicts


35
30
25
20
15
10
5
0
0
10
20
30
40
50
60
70



1200
1000
800
600
400
200
0
0
50
100
150
200
250
300
350
400

Glucose Restarts in a Nutshell

```
bool restarting () {
  return conflicts >= limit &&
     average_RECENT_lbds () > 1.25 * average_ALL_lbds ();
}

void glucose_restart () {  // same as static_uniform_restart
  restarts++;
  limit = conflicts + 50;
  backtrack (0);
}
```

- glucose level (LBD) of learned clause:
  - number of different decision levels in a learned clauses
  - calculated at the point the clause is learned during conflict analysis
- last 50 LBDs are stored and considered recent (explicit LBD queue)
- total average of all LBDs is simply `sum_lbd / conflicts`
- for discussion of *blocking restarts* since Glucose 2.1 see the paper

Glucose uses *simple moving average* (*SMA*) for the average of recent LBDs and *cumulative moving average* (*CMA*) for the the average of all LBDs and

simple $\quad SMA(n,w) = \frac{1}{w} \cdot (t_n + t_{n-1} + \ldots + t_{n-w+1})$ with $n \geq w \geq 1$

cumulative $\quad CMA(n) = SMA(n,n)$

$$CMA(n) = CMA(n-1) + \frac{t_n - CMA(n-1)}{n}$$

$$SMA(n,w) = SMA(n-1,w) + \frac{t_n}{w} - \frac{t_{n-w}}{w}$$

requires $\quad SMA(n,50) > 1.25 \cdot CMA(n)$ $\quad$ to restart

and 50 conflicts have passed

we suggest to use *EMA*s instead of the "fast" *SMA* and/or "slow" *CMA*

exponential $\mathit{EMA}(n,\alpha) = \alpha \cdot t_n + (1-\alpha) \cdot \mathit{EMA}(n-1,\alpha)$ with $0 < \alpha < 1$ $a \approx \frac{2}{1+w}$

current estimate

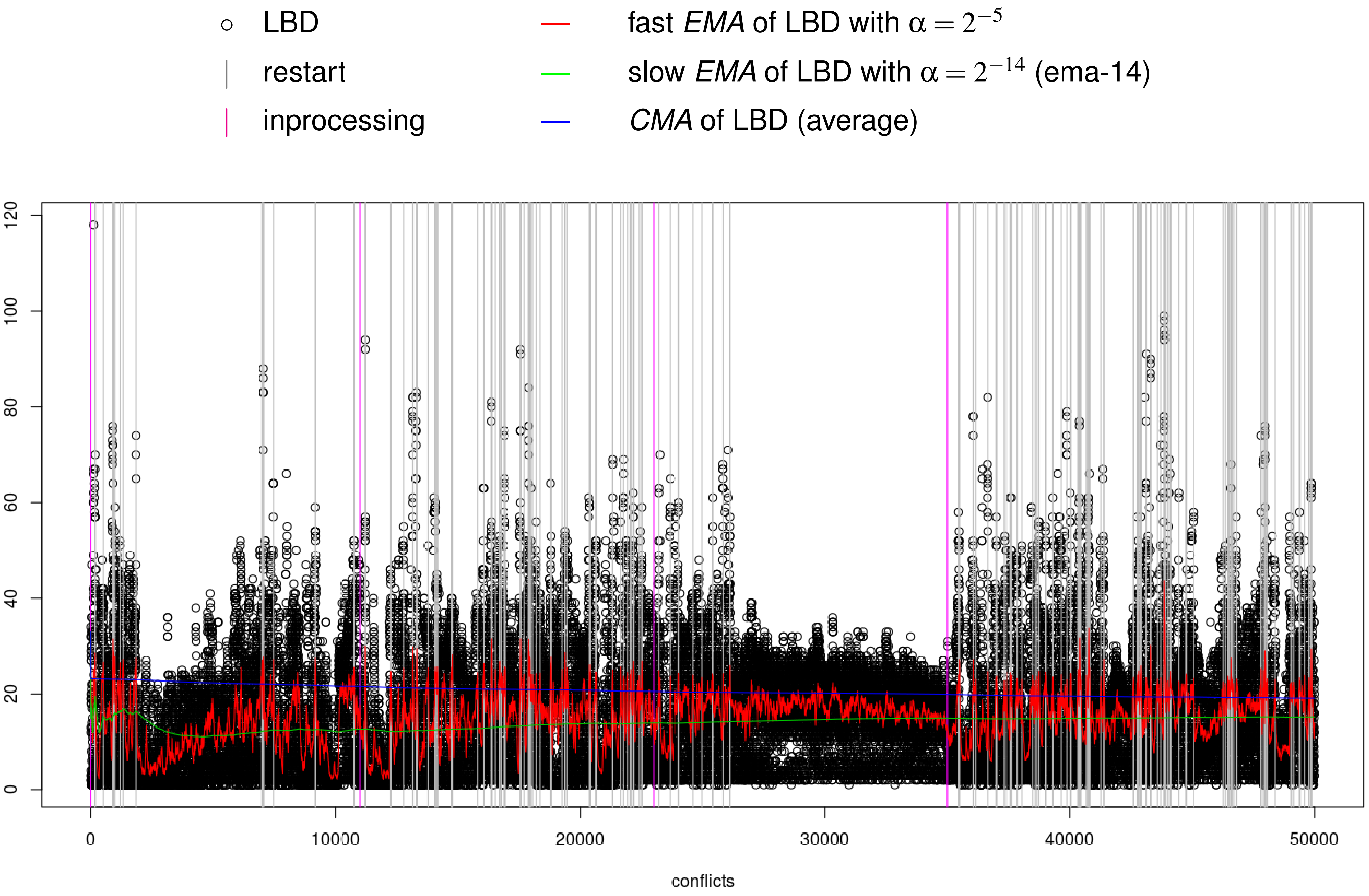
alternative $\mathit{EMA}(n,\alpha) = \mathit{EMA}(n-1,\alpha) + \alpha \cdot (t_n - \mathit{EMA}(n-1,\alpha))$

next estimate difference/error

to restart version *average* requires $\mathit{EMA}(n,2^{-5}) > 1.25 \cdot \mathit{CMA}(n)$

to restart version *ema-14* requires $\mathit{EMA}(n,2^{-5}) > 1.25 \cdot \mathit{EMA}(n,2^{-14})$

and again in both cases that a certain number of conflicts say 50 have passed



LBD
restart
inprocessing
fast EMA of LBD with $\alpha = 2^{-5}$
slow EMA of LBD with $\alpha = 2^{-14}$ (ema-14)
CMA of LBD (average)
0
20
40
60
80
100
120
0
10000
20000
30000
40000
50000
conflicts

Comparing EMAs with SMA and CMA

solver	Glucose 4.0			Lingeling ba2							
restarts	*ss*	*es*	*ee*	*avg*	*e8*	*e10*	*e12*	*e14*	*e16*	*e18*	*e20*
tot	163	163	165	178	167	170	180	**181**	180	177	171
sat	72	73	76	83	80	78	**86**	**86**	**86**	82	77
uns	91	90	89	**95**	87	92	94	**95**	94	**95**	94
avgc	192	166	167	145	230	204	195	186	172	147	108

Glucose 4.0 column *ss* correspond to the original Glucose version

column *es* to adding EMAs for only forcing restarts

column *ee* includes using EMA for blocking restarts too

column *avg* is Lingeling version *average* of Glucose version *ee*

columns *eX* correspond to Lingeling versions *ema*-X
using a slow EMA with $\alpha = 2^{-X}$ instead of CMA

```
double fast, slow;
...

bool analyze () {
  int lbd;
  ...
  slow += (lbd - slow)/(double)(1<<14);
  fast += (lbd - fast)/(double)(1<<5);
  ...
}

bool restarting () {
  return conflicts > limit && fast > 1.25 * slow;
}
```

fast 64-bit fixed point implementation avoiding floating point

inspired by Donald Knuth's implementation of our agility metric

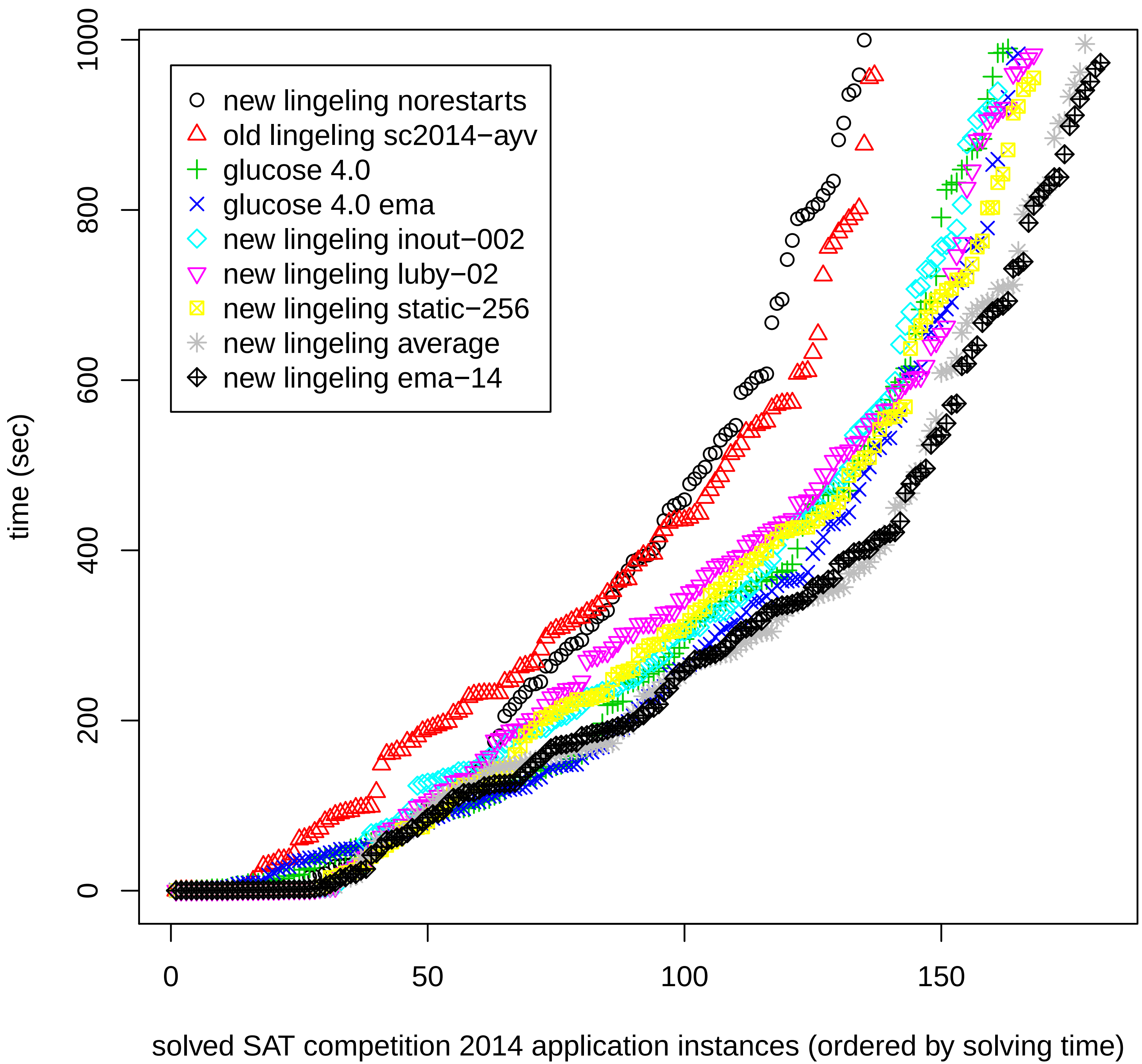
```
long fast, slow;                    // assume (sizeof (long) == 8);
...                                 // initialization code skipped ...

bool analyze () {
  int lbd;                          // assume (sizeof (int) == 4);
  ...
  fast -= fast >> 5;
  fast += lbd << (32 - 5);
  slow -= slow >> 14;
  slow += lbd << (32 - 14);
  ...
}

bool restarting () {
  return conflicts > limit && fast / 125 > slow / 100;
}
```

Conclusion

- data and source: http://fmv.jku.at/evalrestart/evalrestart.7z
- optimal restart interval varies with benchmark bucket
  - for miters fast restarts essential
  - for crypto benchmarks longer intervals necessary
  - disabling restarts completely is bad
  - Glucose restarts superior to Luby style
- presented an EMA variant of the Glucose restart scheme
  - simpler model, simpler to implement
  - similar performance (slightly faster)
- future work
  - how to improve blocking of restarts
  - restart intervals still not optimal: really need machine learning?
  - finally cross-fertilize ideas from *SAT and Stock Market Analysis*
    originally proposed title for this paper



new lingeling norestarts
old lingeling sc2014−ayv
glucose 4.0
glucose 4.0 ema
new lingeling inout−002
new lingeling luby−02
new lingeling static−256
new lingeling average
new lingeling ema−14
time (sec)
0
200
400
600
800
1000
0
50
100
150
solved SAT competition 2014 application instances (ordered by solving time)