GlueMiniSat 2.2.7j: On-The-Fly Lazy Clause Simplification

Hidetomo NABESHIMA*

Koji IWANUMA*

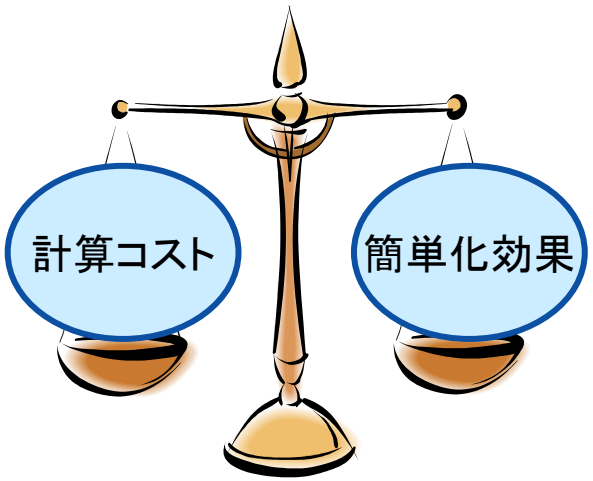
Katsumi INOUE**

* University of Yamanashi

** National Institute of Informatics

はじめに

- GlueMiniSat 2.2.7j
  - 特徴：軽量かつ動的な節簡単化技術
    - Probing [Lynce+ 03] と自己包摂融合を動的に実行
    - 計算コストはほぼ O(1)
    - ソルバプロセスの至る所で頻繁に簡単化を試みる

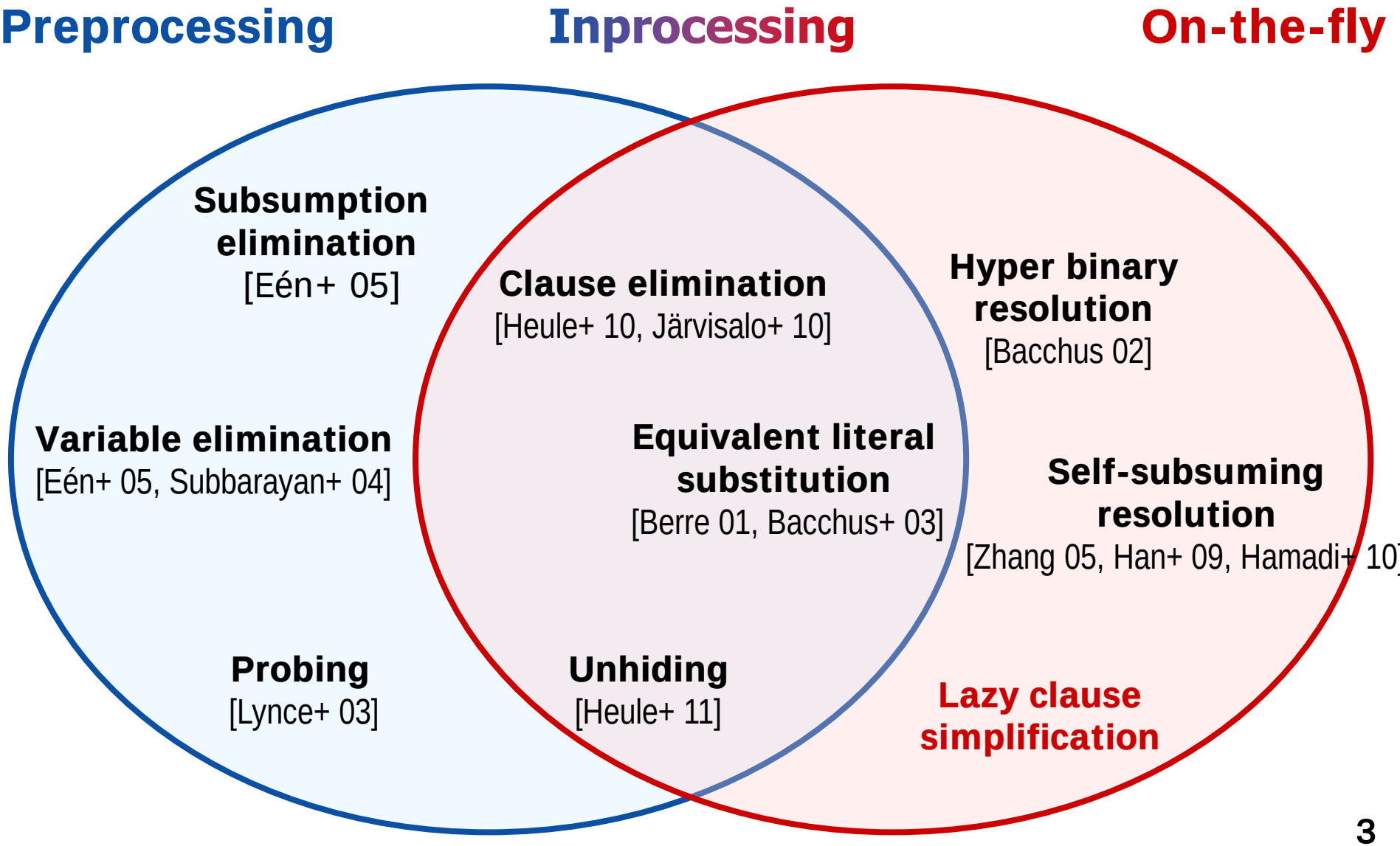


簡単化技術のトレードオフ

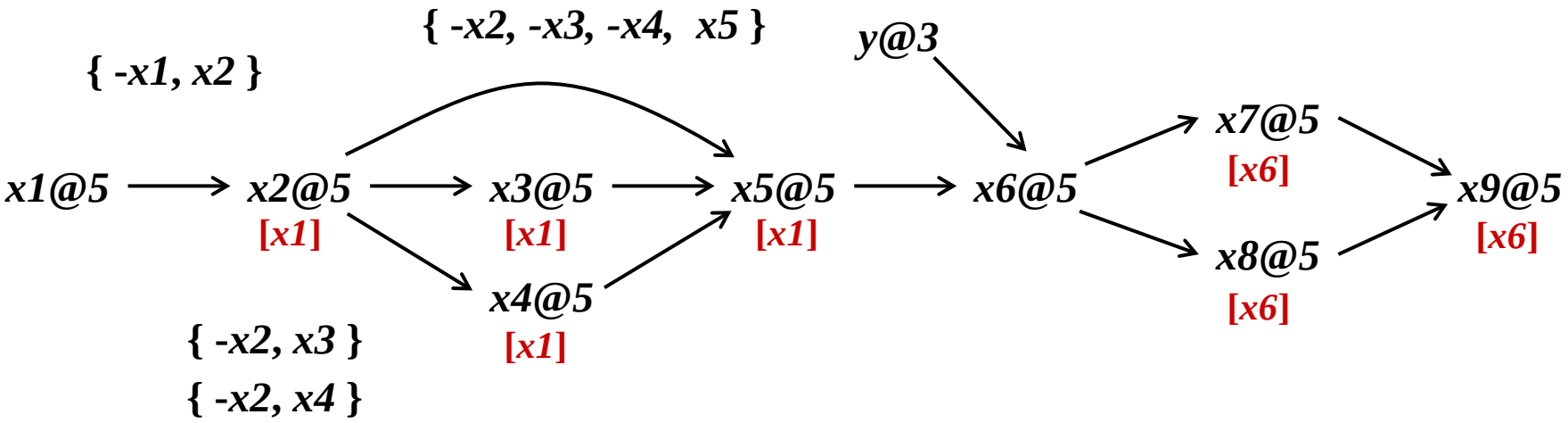
✓ 極端に計算コストを抑えた簡単化技術

✓ 簡単化は不完全だが，頻繁に実行

SAT の簡単化技術


Preprocessing
Inprocessing
On-the-fly
Subsumption elimination
[Eén+ 05]
Variable elimination
[Eén+ 05, Subbarayan+ 04]
Probing
[Lynce+ 03]
Clause elimination
[Heule+ 10, Järvisalo+ 10]
Equivalent literal substitution
[Berre 01, Bacchus+ 03]
Unhiding
[Heule+ 11]
Hyper binary resolution
[Bacchus 02]
Self-subsuming resolution
[Zhang 05, Han+ 09, Hamadi+ 10]
Lazy clause simplification

BCP から Bin Resolvents を抽出



- SAT ソルバは BCP を頻繁に実行（実行時間の 80 ~ 90%）
- 各リテラルに対し，dominator を O(1) で検出 [Han+ 11]

$x1 \rightarrow x2$
$x1 \rightarrow x3$
$x1 \rightarrow x4$
$x1 \rightarrow x5$
$x6 \rightarrow x7$
$x6 \rightarrow x8$
$x6 \rightarrow x9$

✓ 最悪の場合 $O(n^2)$ 存在（n: 変数の数）

✓ どのように保持・管理するか？

Bin Resolvents の管理

- [Han 11] では，節 DB に追加
  - 問題によっては多数生成され，BCP 速度が低下
  - 重複する節や冗長な節の除去が必要
- 本手法では，各リテラルに対し，dominator を1つだけ保存
  - 配列 **pre[]** を用意（サイズはリテラル数）
  - **pre[q]** は **q** の dominator を表す（**pre[q]** → **q**）
  - **pre[q]** の値は，**q** が単位伝搬されるたびに，その dominator で更新

pre[q] の値は頻繁に更新される

✓評価実験では，1リテラルあたり約2000回更新（1200秒制限）

✓よって，すべての binary resolvents を保持しなくても良い（後述）

On-The-Fly Probing

- 前処理技術 Probing [Berre 01, Lynce 03] を on-the-fly で実行
- Necessary Assignment Probing

$$\phi \models x \rightarrow y \quad \wedge \quad \phi \models \neg x \rightarrow y \quad \Rightarrow \quad \phi \models y$$

if (pre_{old}[y] = ¬pre_{new}[y]) y = true

- Equivalent Variable Probing

$$\phi \models x \rightarrow y \quad \wedge \quad \phi \models \neg x \rightarrow \neg y \quad \Rightarrow \quad \phi \models x \leftrightarrow y$$

if (pre_{old}[y] = ¬pre[¬y] or pre_{new}[y] = ¬pre[¬y]) x = y

- (Binary) Clause Probing

$$\phi \models x \vee z \quad \wedge \quad \phi \models x \rightarrow y \quad \wedge \quad \phi \models z \rightarrow y \quad \Rightarrow \quad \phi \models y$$

if (pre_{old}[y] == ¬pre[pre_{new}[y]] or
pre_{new}[y] == ¬pre[pre_{old}[y]) y = true

pre[y] が更新されると即座に実行．計算コストは O(1)

On-the-fly Self-Subsuming Resolution

$$\begin{array}{rl} C: & \{w_1, w_2, x_1, \cdots, x_j, \cdots, x_k\} \\ & \{w_i, \neg x_j\} \\ \hline C': & \{w_1, w_2, x_1, \cdots, x_{j-1}, x_{j+1}, \cdots, x_k\} \end{array}$$

if (pre[wi] == xj or pre[¬xj] == ¬wi) remove xj

- 節の各リテラルペアについてチェックすると O(k^2)
- 監視リテラルと非監視リテラルのペアに限定 O(k)
- CDCL ソルバにおける節内のリテラル走査に便乗 O(1)
  - Unit propagation, conflict analysis, removal of satisfied clauses
- 獲得した学習節の簡単化のため，監視リテラルと非監視リテラルのペアに限定して簡単化 O(k)

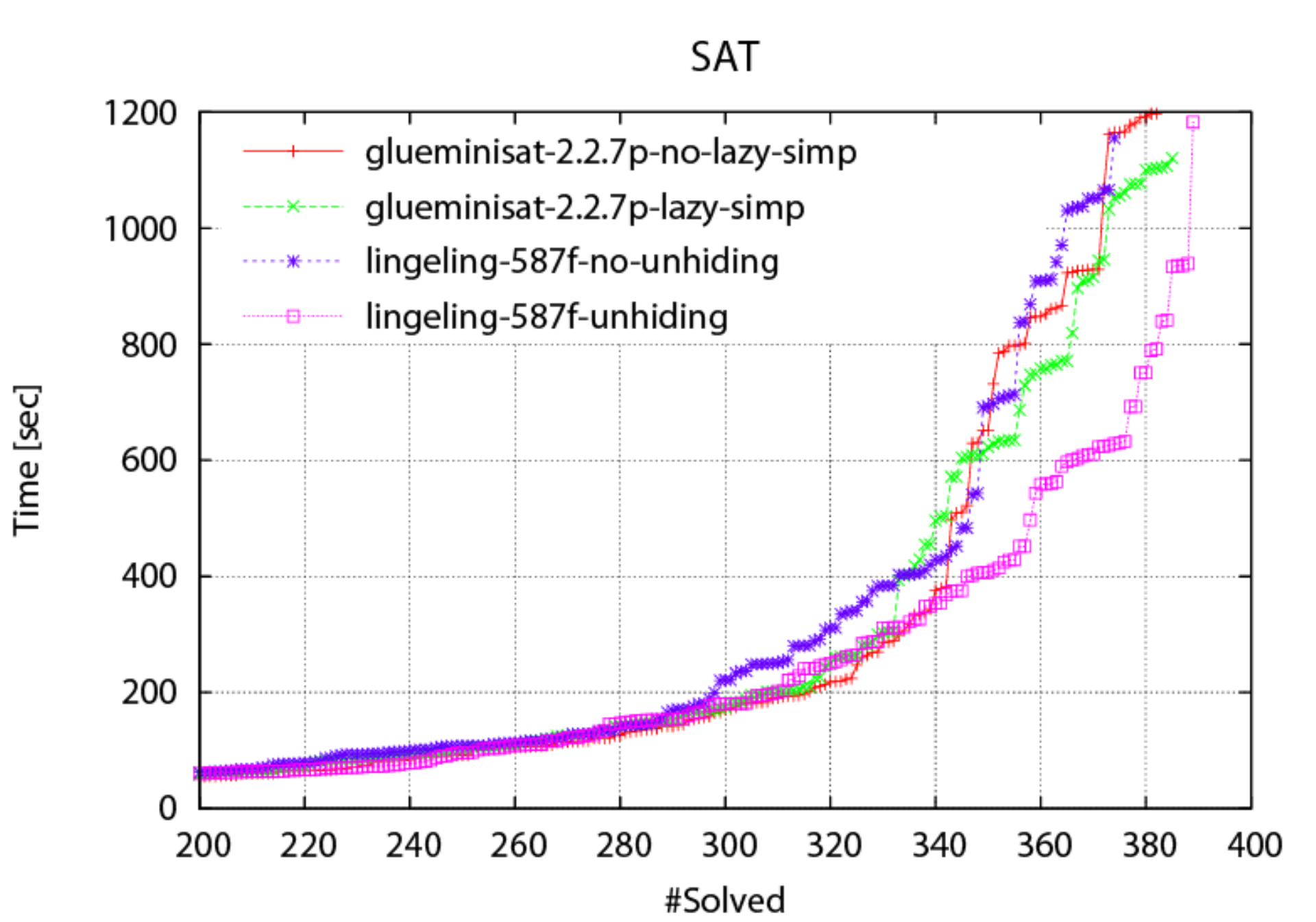
評価実験

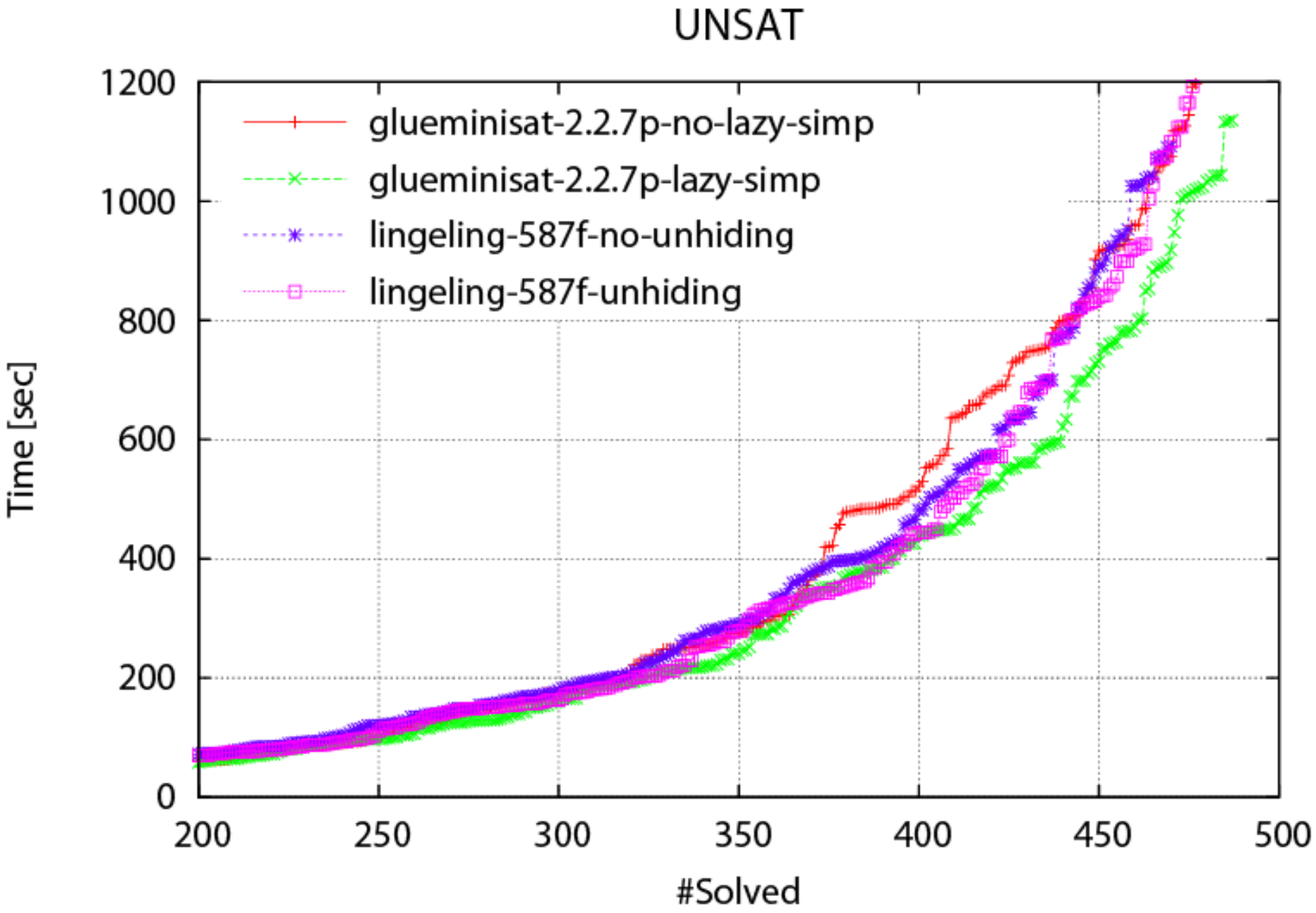
- Solvers
  - Glueminisat 2.2.7p with/without lazy simp
  - Lingeling 587f with/without unhiding [Heule+ 11]
    - **HLE, HTE, HBR, FLE, ELS, TRD**
    - Linear time in total number of literals
- Benchmarks
  - SAT 2009 Competition Application 292 問
  - SAT 2011 Competition Application 300 問
  - SAT Challenge 2011 Application 600 問
- Environment
  - Core i7 (2GHz) with 8GB memory
  - gcc 4.2
  - Timeout: 1200 sec / instance

実験結果

	GlueMiniSat 2.2.7		Lingeling 587f	
	w/o lazy simp	with lazy simp	w/o unhiding	with unhiding
SAT 2009 Competition	193 (73 + 120)	196 (74 + 122)	194 (75 + 119)	198 (79 + 119)
SAT 2011 Competition	180 (81 + 99)	184 (82 + 102)	164 (76 + 88)	169 (79 + 90)
SAT Challenge 2012	488 (229 + 259)	494 (230 + 264)	488 (224 + 264)	500 (232 + 268)
Total	861 (383 + 478)	874 (386 + 488)	846 (375 + 471)	867 (390 + 477)

- Lazy clause simplification は UNSAT に効果的 (+10問)
  - On-the-fly probing は約 10% の変数の値を確定
  - On-the-fly SSR は総リテラル数の 1% 程度を除去
- Unhiding は SAT に効果的 (+15問)
- Glueminisat は SAT 2011 Application をベースにチューニング
  - SAT 2011 の結果は良いが，その他の伸びがイマイチ


SAT
glueminisat-2.2.7p-no-lazy-simp
glueminisat-2.2.7p-lazy-simp
lingeling-587f-no-unhiding
lingeling-587f-unhiding
Time [sec]
0
200
400
600
800
1000
1200
200
220
240
260
280
300
320
340
360
380
400
#Solved


UNSAT
glueminisat-2.2.7p-no-lazy-simp
glueminisat-2.2.7p-lazy-simp
lingeling-587f-no-unhiding
lingeling-587f-unhiding
Time [sec]
#Solved
0
200
400
600
800
1000
1200
200
250
300
350
400
450
500

SAT 2013 Competition

- 以下の track に参加
  - Application SAT
  - Application certified UNSAT
  - Application SAT+UNSAT
  - Hard-combinatorial SAT
  - Hard-combinatorial certified UNSAT
  - Hard-combinatorial SAT+UNSAT
  - Open track

- Certified UNSAT では UNSAT 判定時に証明が必要
  - Resolution trace format in 2005
    - 反駁に至る導出木を出力
  - Reverse Unit Propagation (RUP) format in 2007 [Allen+ 07]

RUP Format

- 求解中に学習した節をすべて出力
  - ただし，各節 C は単位伝搬のみで導出可能であること
  - 検証器は以下をチェック

$$\Sigma_{\mathrm{org}} \wedge \Sigma_{\mathrm{learnt}} \wedge \neg C \models_{UP} \bot$$

  - もし上記を満たせば Σ_{learnt} に C を追加
- 最終的に空節が導出できれば（$\Sigma_{\mathrm{org}} \wedge \Sigma_{\mathrm{learnt}} \models_{UP} \bot$）証明終了

- 実装は比較的容易だが，検証に多数の単位伝搬が必要となるため，resolution trace format と比較して検証に時間がかかる
  - Application certified UNSAT には 11 ソルバが参加（ほとんどが RUP）
  - 一方，Application SAT + UNSAT には 29 ソルバが参加
- On-the-fly lazy simplification との相性は悪い（binary resolvents を覚えておく必要がある）．一部の機能を無効化して参加

SAT 2013 Competition Results

	Application			Hard combinatorial			Open		
	Gold	**Silver**	**Bronze**	**Gold**	**Silver**	**Bronze**	**Gold**	**Silver**	**Bronze**
SAT+ UNSAT	Lingeling aqw	Lingeling 587f	ZENN 0.1.0	BreakID Glucose	gluebit_ clasp 1.0	glucose 2.3	CSHCpar 8	MIPSat	GlucoRed +March r531
SAT	Lingeling aqw	ZENN 0.1.0	satUZK 48	glucose 2.3	gluebit_ clasp 1.0	BreakID Glucose	-	-	-
Cert. UNSAT	glucose 2.3	glue minisat	Riss3g	Riss3g	glucose 2.3	forl drup-nocaches tamp	-	-	-

- Application SAT 19th (31 solvers)
- Application certified UNSAT 2nd (11 solvers)
- Application SAT+UNSAT 12th (29 solvers)
- Hard-combinatorial SAT 18th (41 solvers)
- Hard-combinatorial certified UNSAT 5th (9 solvers)
- Hard-combinatorial SAT+UNSAT 5th (35 solvers)
- Open track 12th (13 solvers)

Application Certified UNSAT

Rank	Solver	#Solved
	Virtual Best Solver (VBS)	106
1	glucose 2.3 (certified unsat)	94
2	glueminisat-cert-unsat 2.2.7j	91
3	Riss3g cert	85
4	forl drup-nocachestamp	84
5	forl drup	84
6	Nigma-DRUP 1.0	78
7	Nigma-NoPB-DRUP 1.0	73
8	minisat_bit_u 1.0	68
9	minisat DRUP DRUP	65
10	Lingeling aqw-drup	48
11	SAT4J Certified SAT COMPETITION 2013	15

Application SAT

Rank	Solver	#Solved
	VBS	147
1	Lingeling aqw	119
2	ZENN 0.1.0	113
3	satUZK 48	110
4	Riss3g 3g	108
5	Lingeling 587f	107
6	CSHCapplLG	106
7	gluebit_lgl 1.0	105
7	MIPSat	105
9	forl nodrup	104
10	glucose 2.3	103
11	glue_bit 1.0	102
11	Solver43b b	102
11	strangenight satcomp11-st	102
11	Riss3g	102
15	Solver43a a	101
16	CSHCapplLC	100
16	BreakIDGlucose 1	100
16	glueminisat 2.2.7j	100
:	:	:

Application UNSAT

Rank	Solver	#Solved
	VBS	141
1	Lingeling aqw	112
2	CSHCapplLC	106
3	Lingeling 587f	105
4	glue_bit 1.0	102
5	CSHCapplLG	99
6	glucose 2.3	98
7	strangenight satcomp11-st	97
7	gluH 1.0	97
9	MIPSat	96
9	glueminisat 2.2.7j	96
11	GlucoRed r531	95
11	relback (not competing) v1.1	95
11	ZENN 0.1.0	95
11	BreakIDGlucose 1	95
15	gluH_simp 1.0	94
16	Riss3g 3g	93
17	Solver43b b	92
18	Solver43a a	90
:	:	:

Application SAT+UNSAT

Rank	Solver	#Solved
	VBS	288
1	Lingeling aqw	231
2	Lingeling 587f	212
3	ZENN 0.1.0	208
4	CSHCapplLC	206
5	CSHCapplLG	205
6	glue_bit 1.0	204
7	MIPSat	201
7	Riss3g 3g	201
7	glucose 2.3	201
10	strangenight satcomp11-st	199
11	gluH 1.0	196
11	glueminisat 2.2.7j	196
13	BreakIDGlucose 1	195
14	Solver43b b	194
14	relback (not competing) v1.1	194
16	forl nodrup	192
16	Riss3g	192
18	gluebit_lgl 1.0	191
:	:	:

まとめ

- On-the-fly lazy clause simplification
  - ほとんどオーバーヘッドのない節簡単化技術
- SAT 2013 Competition
  - Application 問題の傾向が変化（SAT Challenge に近い？）

	SAT 2011 Application			SAT 2013 Application		
	Gold	**Silver**	**Bronze**	**Gold**	**Silver**	**Bronze**
SAT+ UNSAT	glucose 2.2	glue minisat	Lingeling 587f	Lingeling aqw	Lingeling 587f	ZENN 0.1.0
SAT	cont-rasat hack	cir_minisat hack	mphasesat64	Lingeling aqw	ZENN 0.1.0	satUZK 48
UNSAT	glue minisat	glucose 2.2	Qute RSat	glucose 2.3	glue minisat	Riss3g

※ Lingeling 587f は SAT Challenge 2012 でも1位（ただし reference solver）

- 今後の予定
  - On-the-fly lazy clause simplification の拡張
  - 多数のベンチマーク問題に対してチューニング