FSE|24

ACM International Conference on the Foundations of Software Engineering

Mon 15 - Fri 19 July 2024 Porto de Galinhas, Brazil, Brazil

Partial Solution Based Constraint Solving Cache in Symbolic Execution

Zhenbang Chen

zbchen@nudt.edu.cn

Joint work with Ziqi Shuai, Kelin Ma, Kunlin Liu, Yufeng Zhang, Jun Sun and Ji Wang

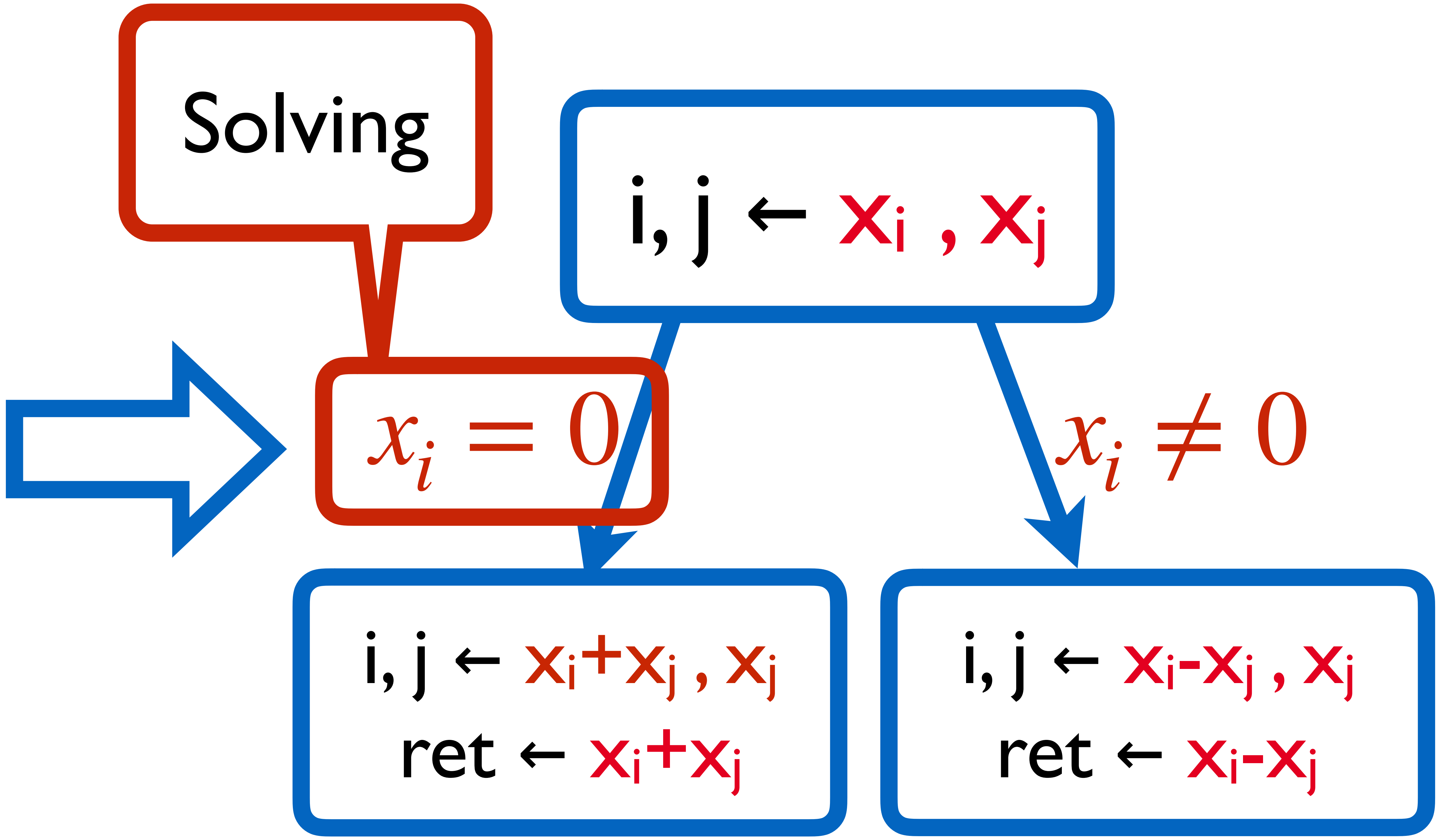
NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY

HUNAN UNIVERSITY

SMU SINGAPORE MANAGEMENT UNIVERSITY

Symbolic Execution

```
int foo(int i, j) {
  if (i == 0) {
    i = i + j
  } else {
    i = i - j
  }
  return i
}
```

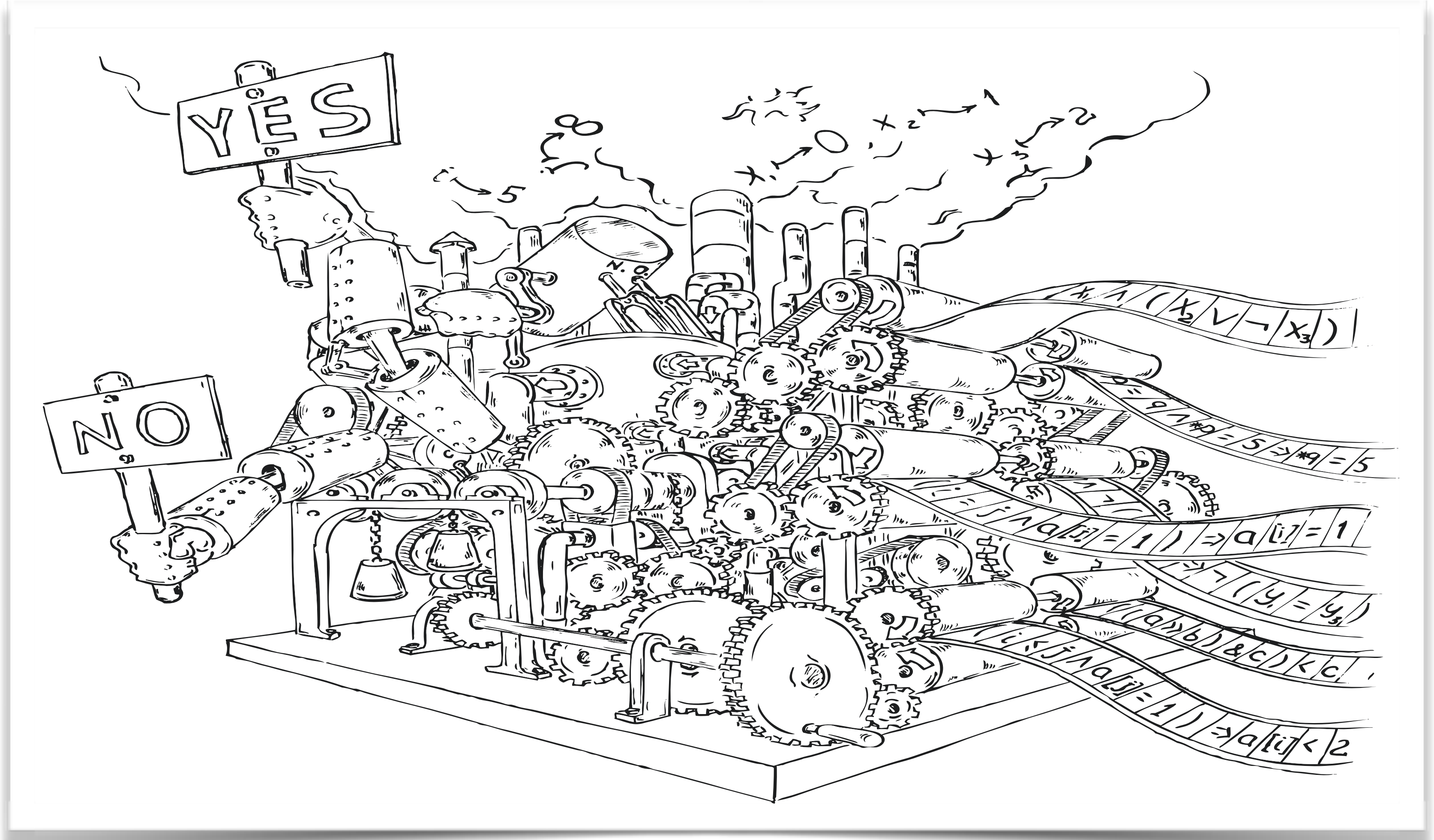


Constraint solving is the enabling technique

Challenges of Symbolic Execution

Path explosion

Constraint Solving

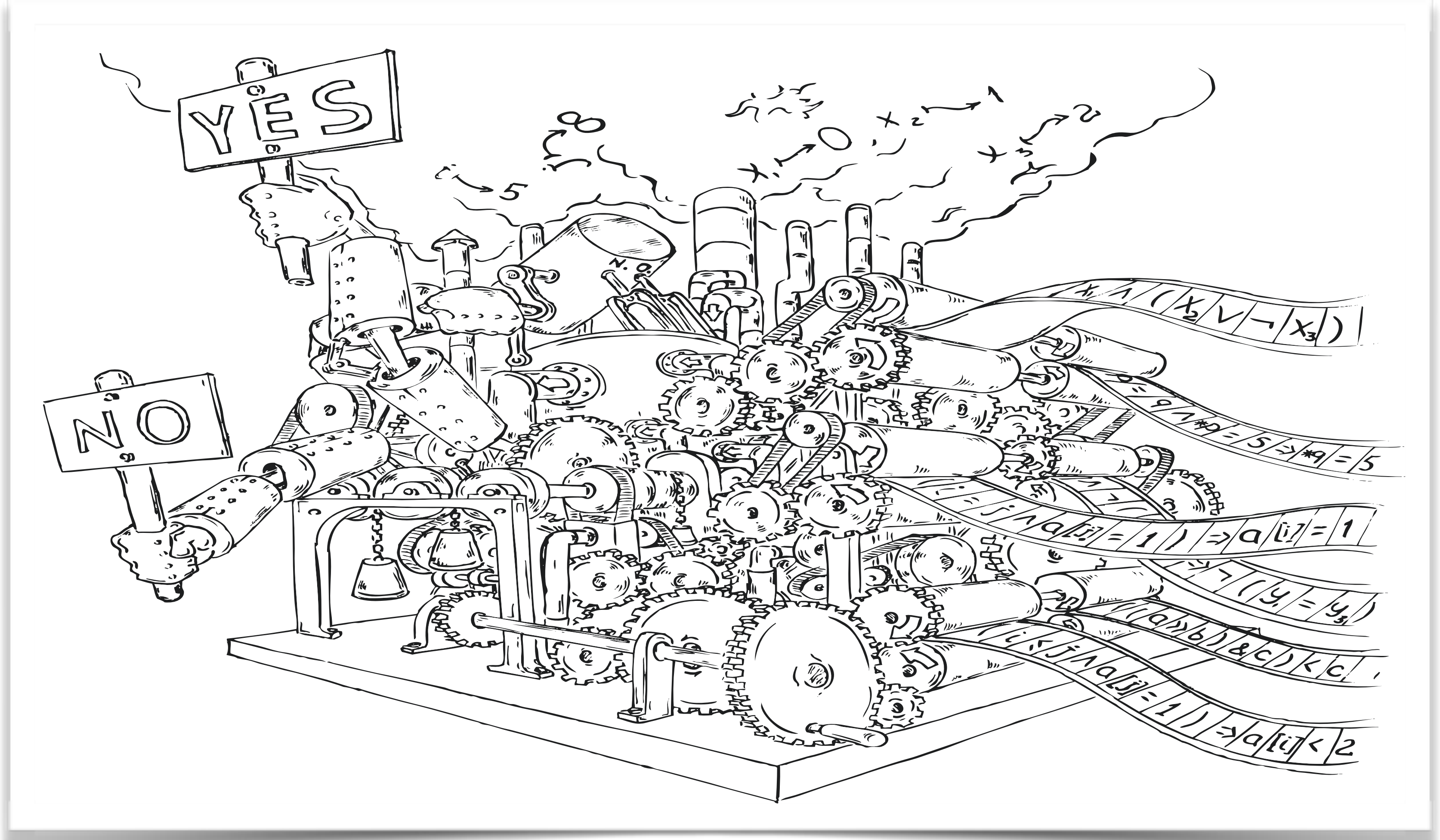


Decision Procedures An Algorithmic Point of View, Second Edition, 2016

This Talk's Target

Path explosion

Constraint Solving



Decision Procedures An Algorithmic Point of View, Second Edition, 2016

Constraint Solving Optimizations

- Optimizing SMT queries
  - Word-level simplifications [OSDI'08][ISSTA'17]
  - Concretization and abstraction [ISSTA'11][Security'15]
- Reducing SMT solver invocations
  - Speculative symbolic execution [ISSRE'12]
  - Caching mechanism [OSDI'08][FSE'12][ISSTA'15]

Caching Mechanism

Constraint Solving Cache

$i < 5 \land i > 10$	$\varnothing$ (**UNSAT**)
$i < 5 \land j > 5$	$\{i = 1, j = 10\}$
$i + j > 5$	$\{i = 1, j = 5\}$
$i = 1 \land j > 10$	$\{i = 1, j = 11\}$

Caching Mechanism

Constraint Solving Cache

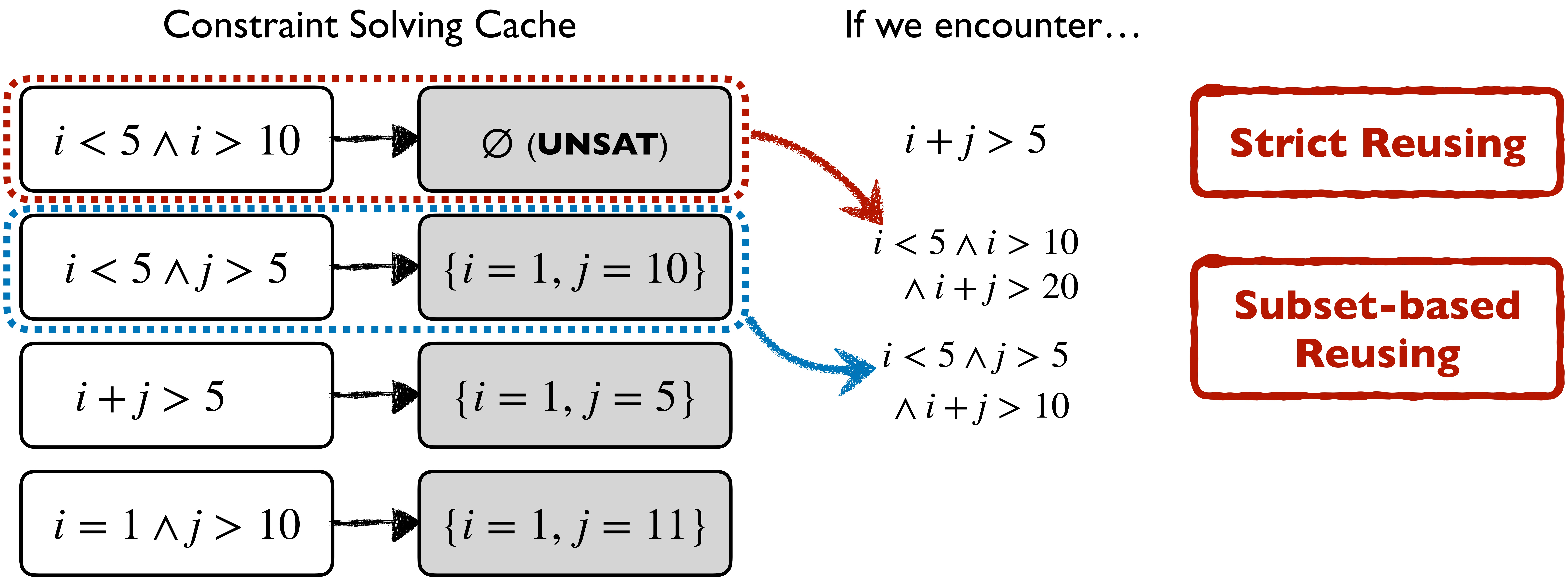
$i < 5 \land i > 10$	$\emptyset$ (**UNSAT**)
$i < 5 \land j > 5$	$\{i = 1, j = 10\}$
$i + j > 5$	$\{i = 1, j = 5\}$
$i = 1 \land j > 10$	$\{i = 1, j = 11\}$

If we encounter…

$i + j > 5$

Strict Reusing

Caching Mechanism



Caching Mechanism

Constraint Solving Cache

$i < 5 \wedge i > 10$ →	$\varnothing$ (**UNSAT**)
$i < 5 \wedge j > 5$ →	$\{i = 1, j = 10\}$
$i + j > 5$ →	$\{i = 1, j = 5\}$
$i = 1 \wedge j > 10$ →	$\{i = 1, j = 11\}$

If we encounter…

$i + j > 5$ — **Strict Reusing**

$i < 5 \wedge i > 10 \wedge i + j > 20$

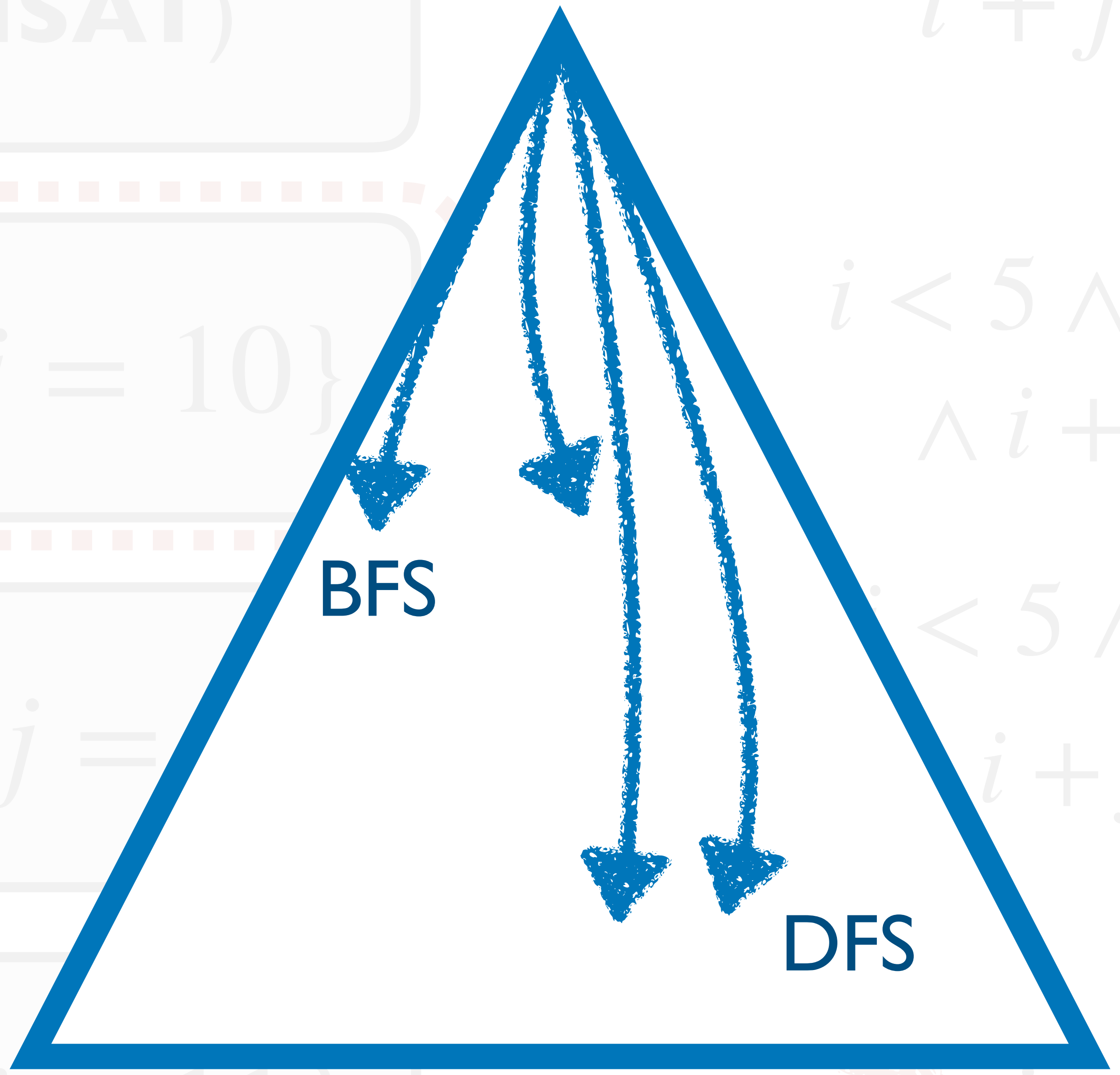
$i < 5 \wedge j > 5 \wedge i + j > 10$ — **Subset-based Reusing**

$i < 5$ — **Superset-based Reusing**

Problem

The effectiveness of the cache depends on many factors

Program Structure



Search Heuristic

Constraint Solver

Related Works

constraint solving cache

Reusing across runs

Green[FSE'12], GreenTrie[ISSTA'15]

Reusing through imprecise matching

Utopia[TSE'21]

Handling address-dependent queries

Trabish et. al[ICST'21]

Related Works

constraint solving cache

Reusing across runs

Green[FSE'12], GreenTrie[ISSTA'15]

Reusing through imprecise matching

Utopia[TSE'21]

Handling address-dependent queries

Trabish et. al[ICST'21]

Improve the usability of existing solutions

Not general

Related Works

CACHE

constraint solving cache

Reusing across runs

Green[FSE'12], GreenTrie[ISSTA'15]

Reusing through imprecise matching

Utopia[TSE'21]

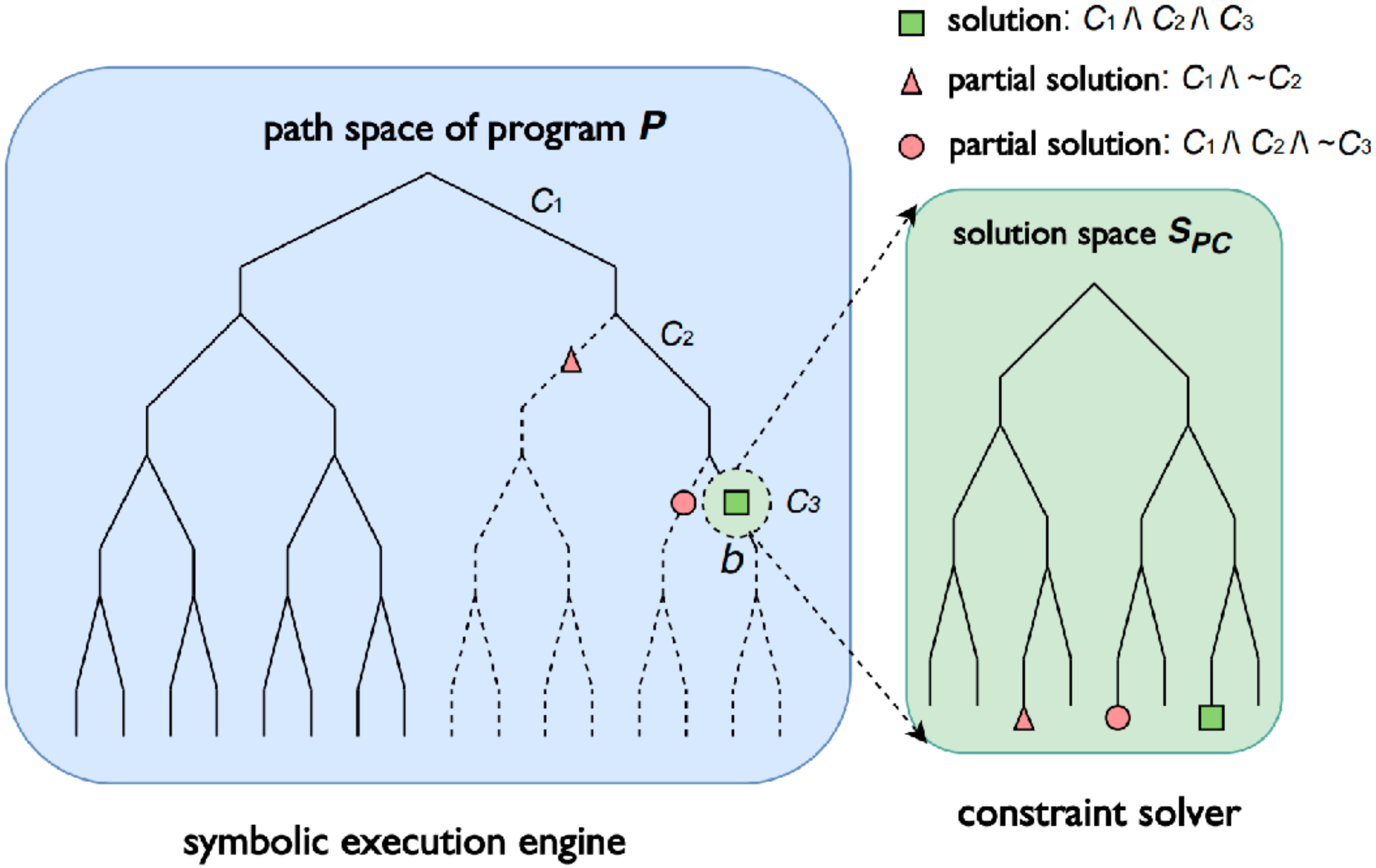
Handling address-dependent queries

Trabish et. al[ICST'21]

Generally improving caching's effectiveness further is still a challenging problem

Our Key Insight (1/2)

- Partial Solutions[1] (PS)
  - Intermediate values in constraint solving
  - Satisfy some sub-constraints



[1] Multiplex Symbolic Execution: Exploring Multiple Paths by Solving Once, ASE'20

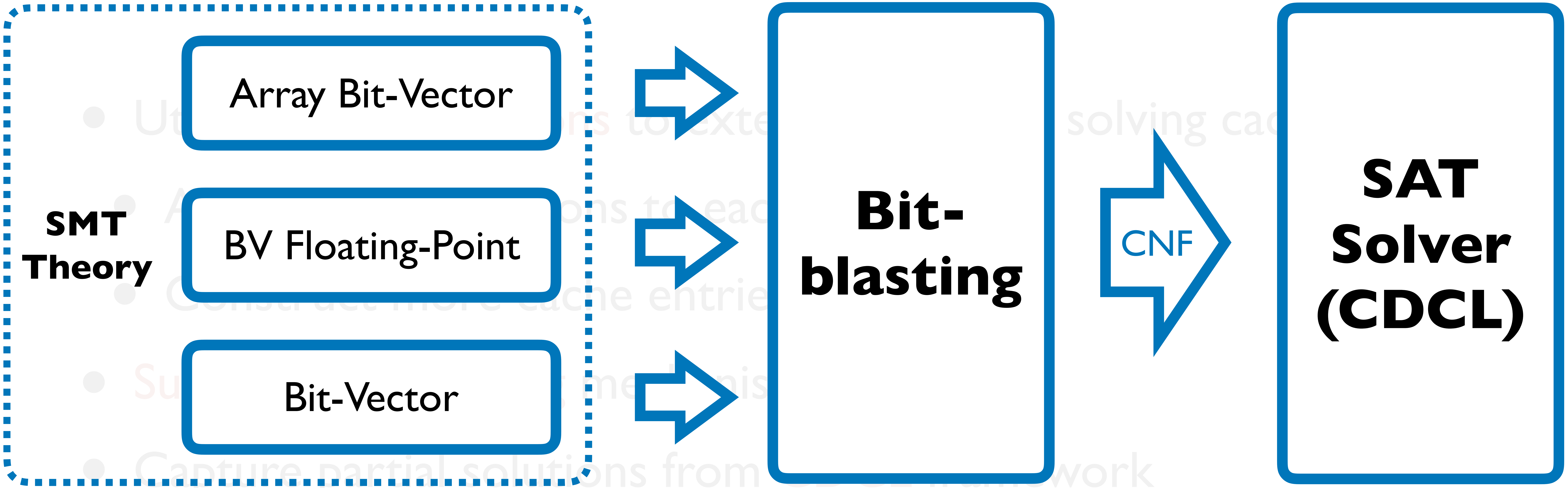
Our Key Insight (2/2)

- Constraint solving often produces lots of partial solutions
  - Abstraction refinement-based array theory solving
  - Optimization-based floating-point solving
  - Simplex-based QF_LIA theory solving
  - Conflict-driven clause learning (CDCL) algorithm
  - …

Key Idea

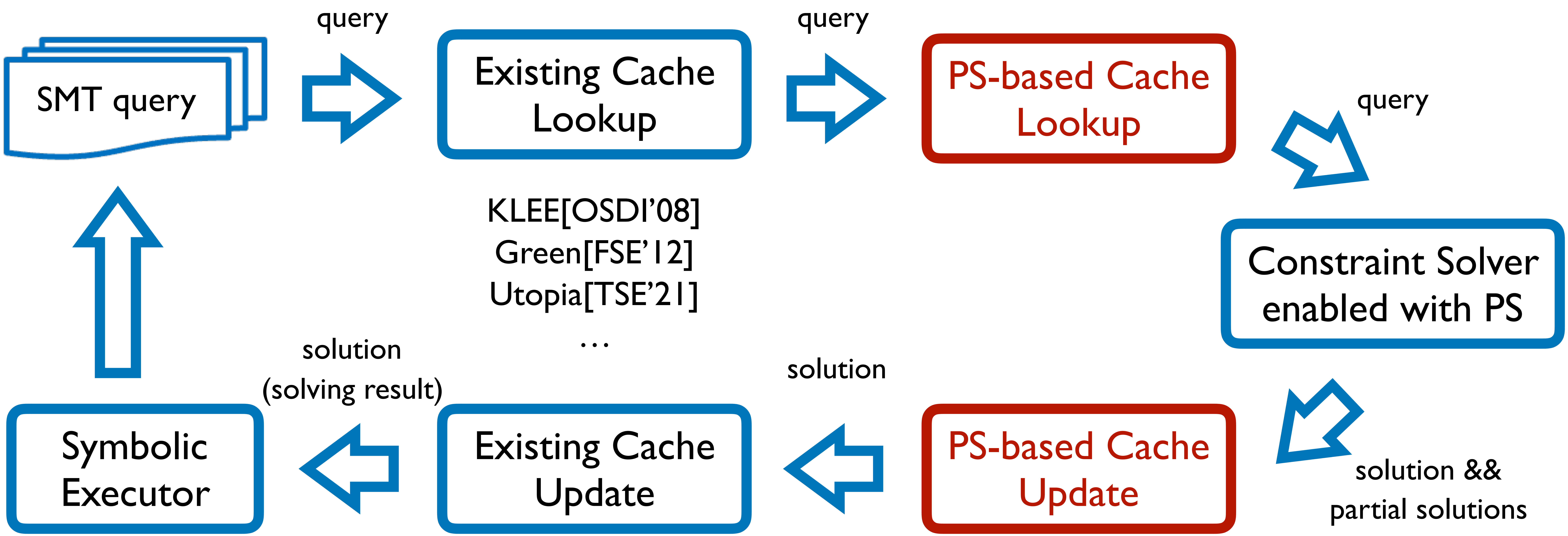
- Utilize partial solutions to expand constraint solving cache
  - Attach more solutions to each cache entry
  - Construct more cache entries
- Subset-based reusing mechanism
- Get partial solutions from CDCL framework

Key Idea

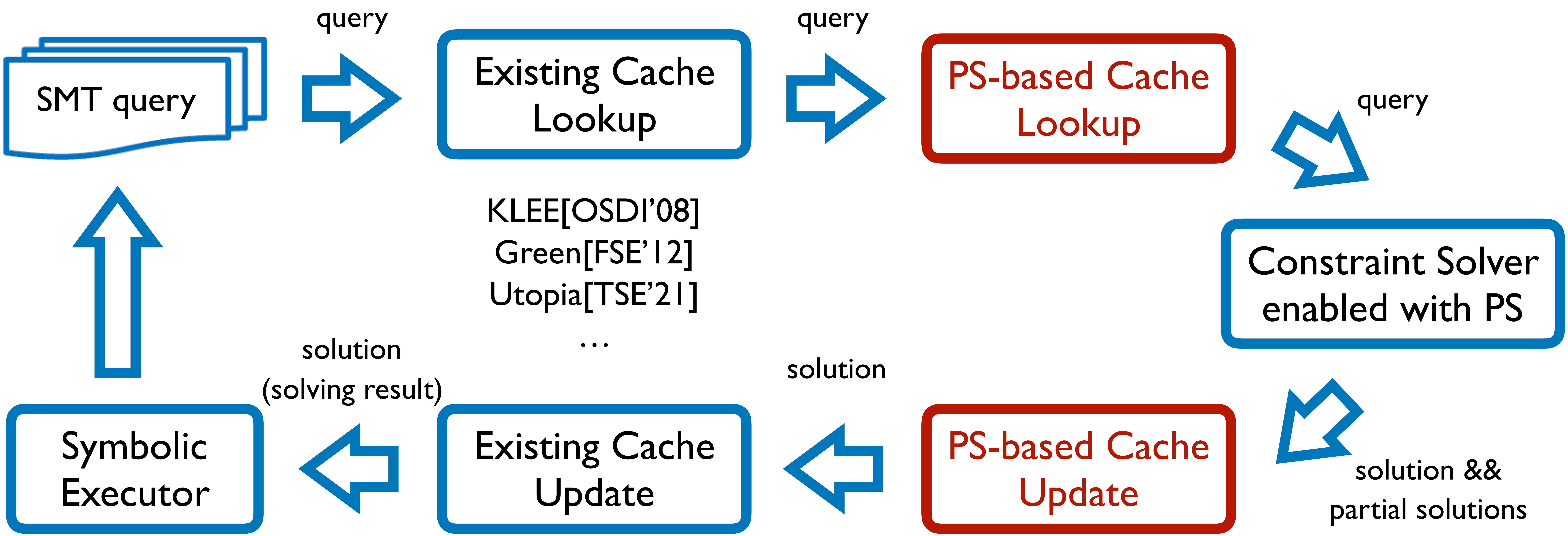


Dominant approaches to SMT rely on calling a CDCL-based SAT solver

Partial Solution-Based Caching



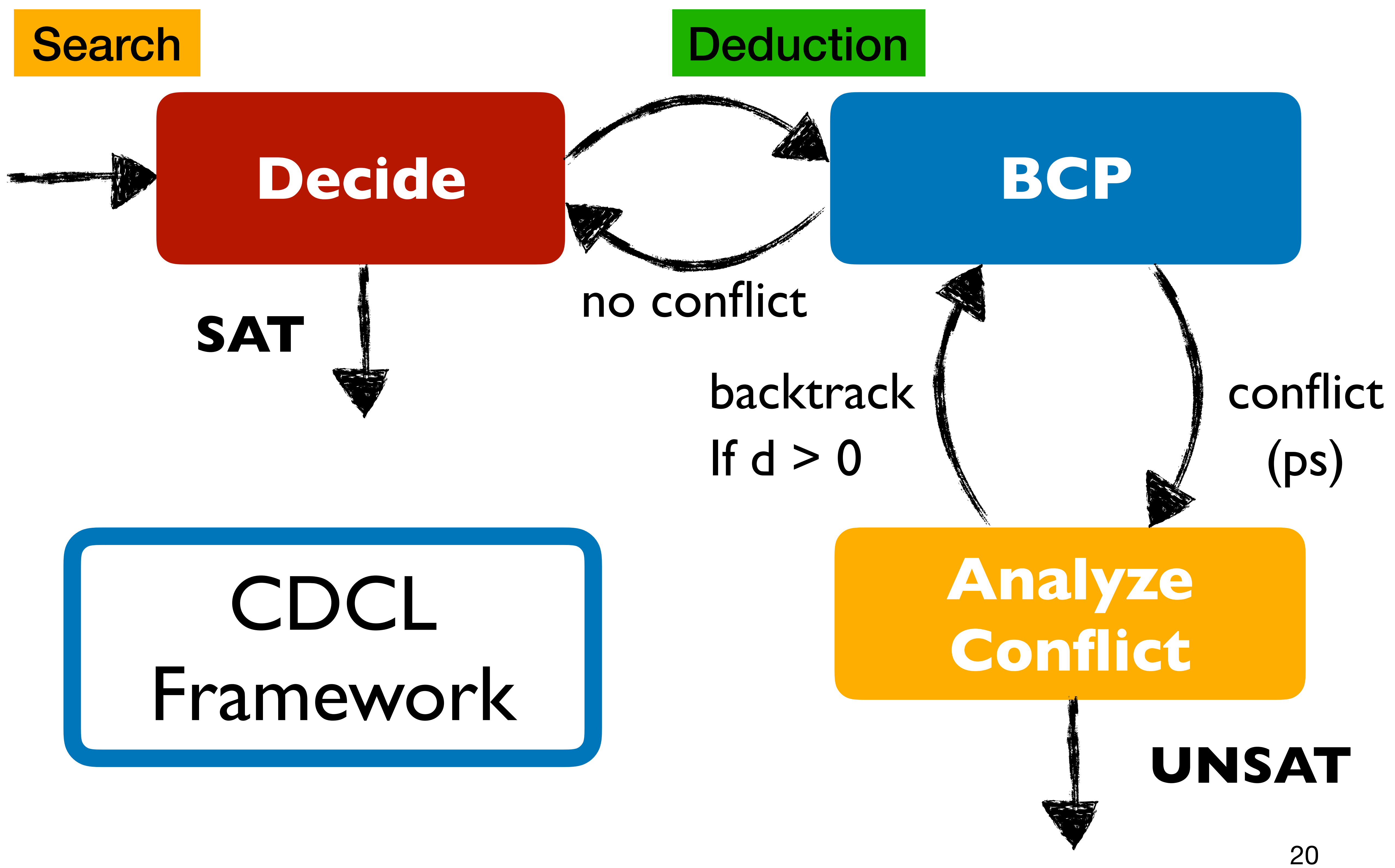
Partial Solution-Based Caching



PS-based caching is orthogonal to existing caching techniques

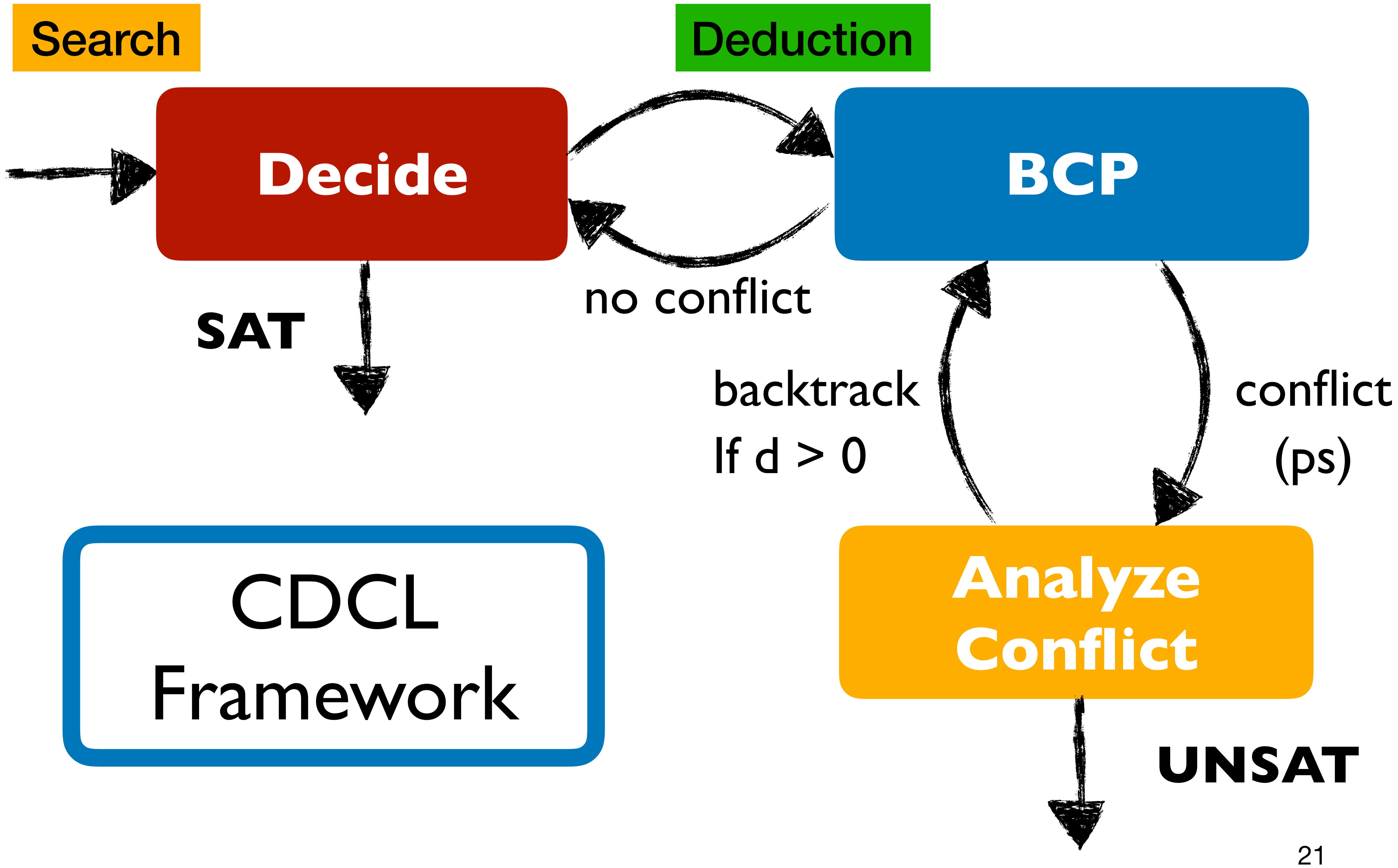
Internals: Build PS

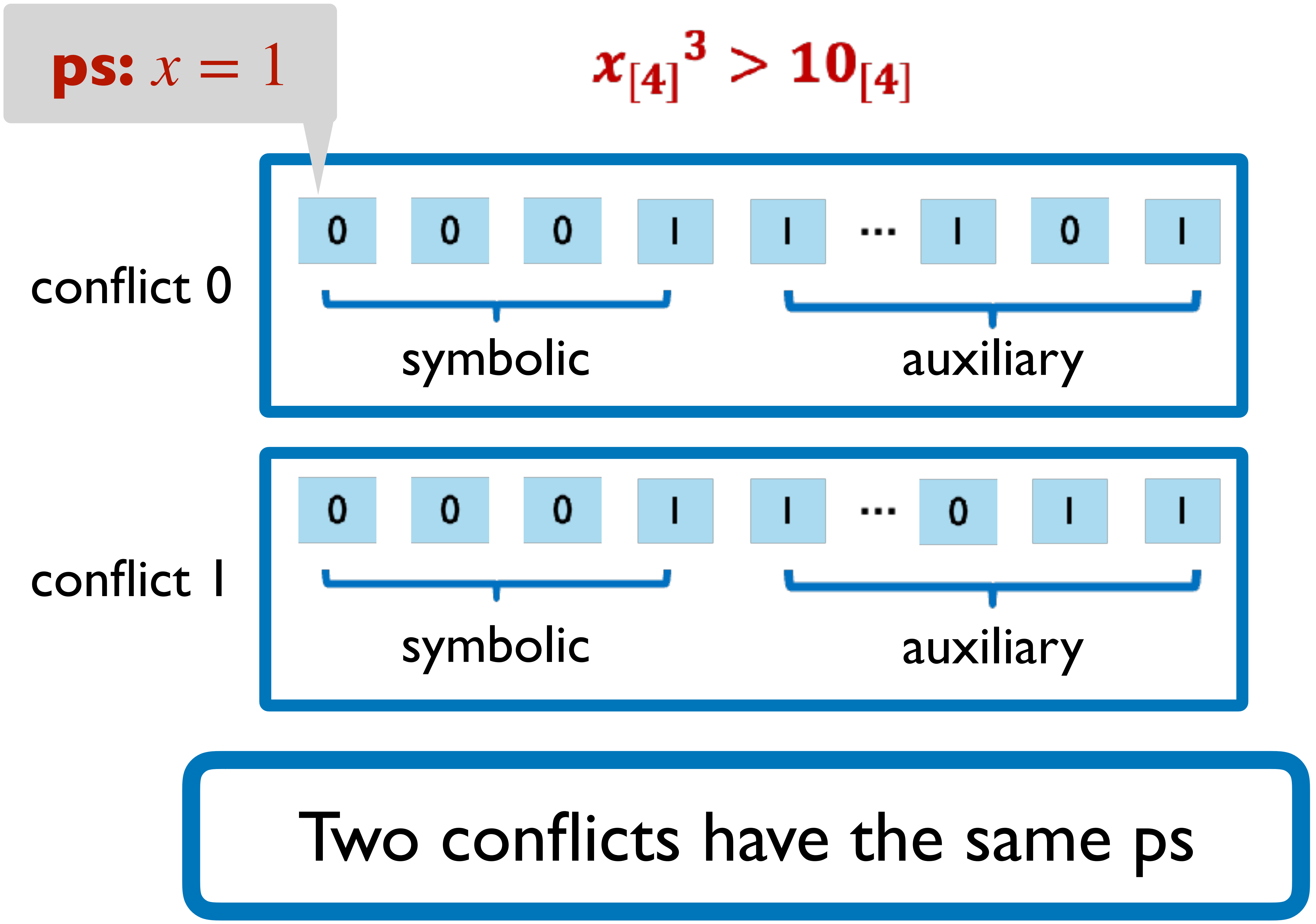
- Build PS based on intermediate assignments causing conflicts



Internals: Build PS

- Build PS based on intermediate assignments causing conflicts
  - Only reserve PS whose symbolic booleans are different

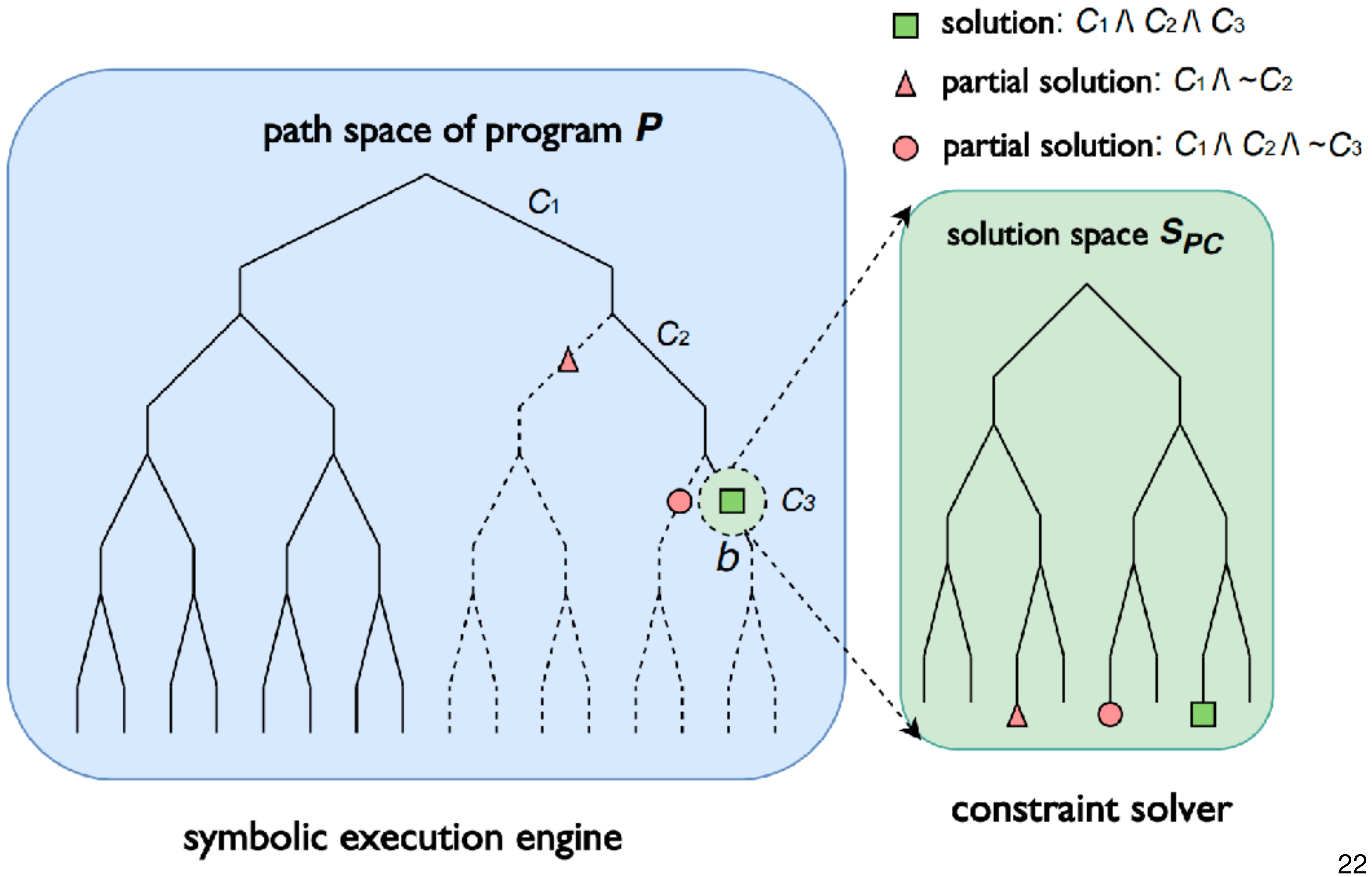




Internals: Expand the Cache

- Construct cache entries: Prefix(φ) ∪ OffThePath(φ)

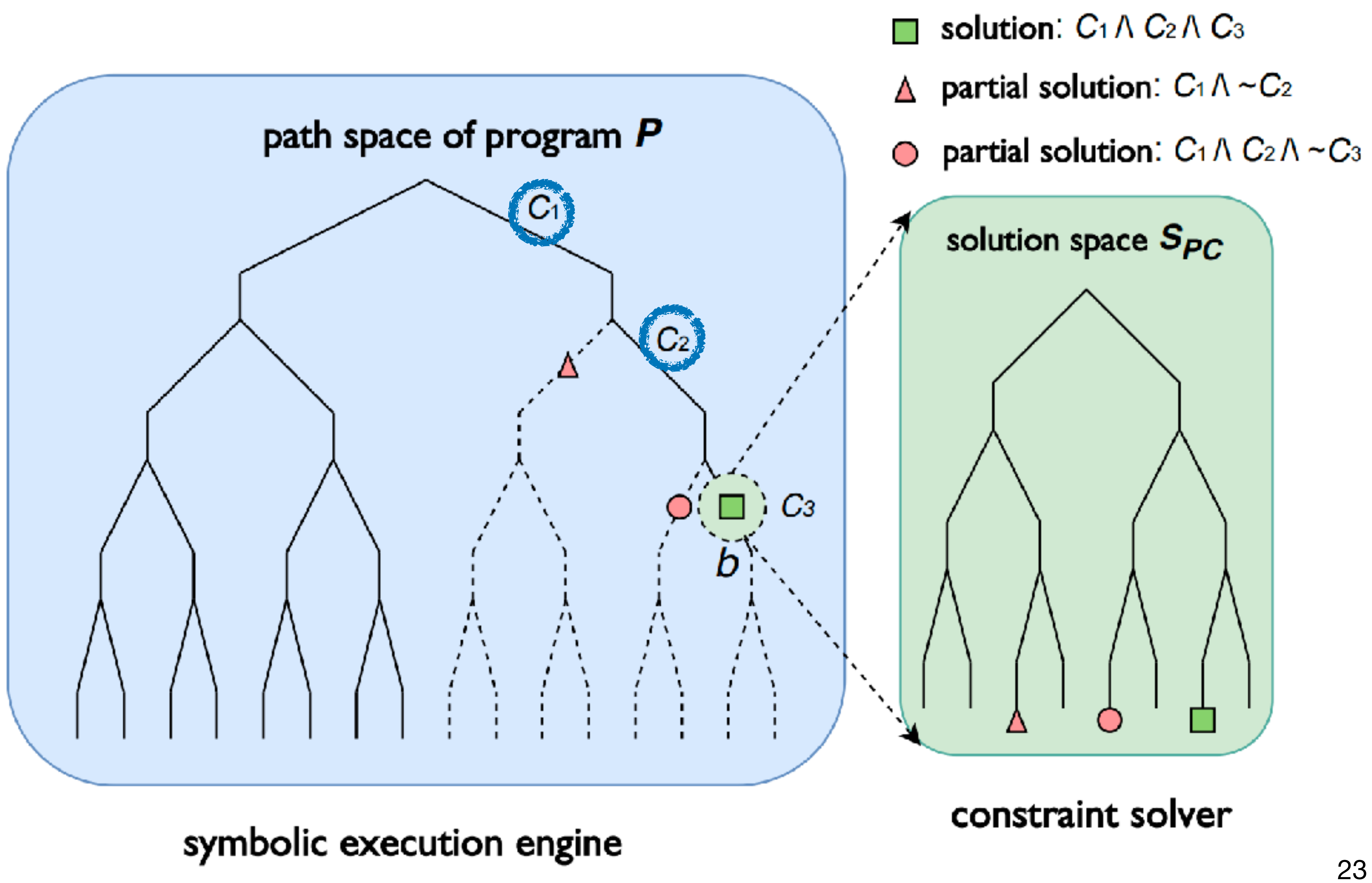
$\varphi = C_1 \wedge C_2 \wedge C_3$



Internals: Expand the Cache

- Construct cache entries: Prefix(φ) ∪ OffThePath(φ)

$\varphi = C_1 \wedge C_2 \wedge C_3$



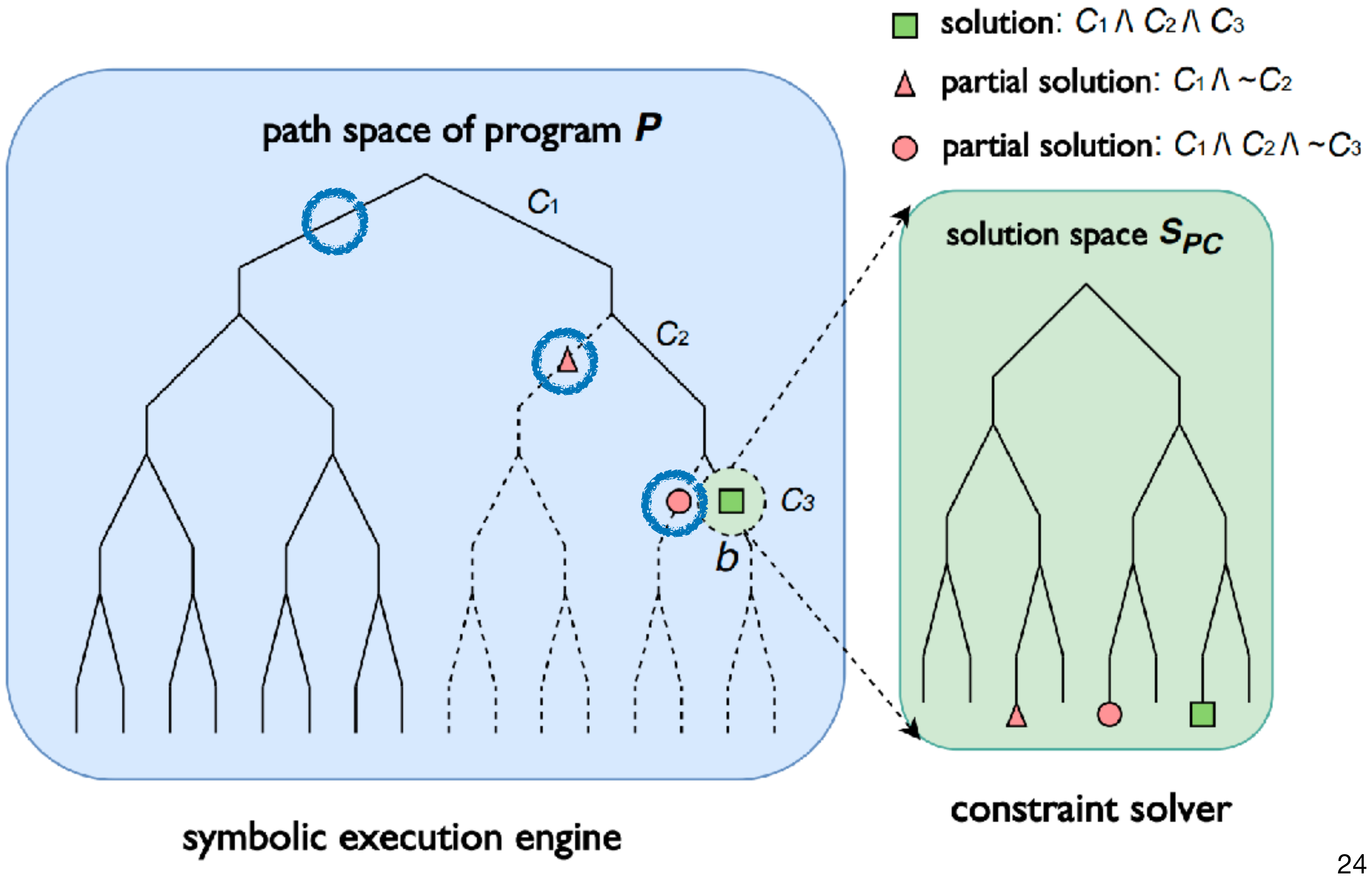
Prefix(φ): all prefixes of φ

$\text{Prefix}(\varphi) = \{C_1, C_1 \wedge C_2\}$

Internals: Expand the Cache

- Construct cache entries: Prefix(φ) ∪ OffThePath(φ)

$\varphi = C_1 \wedge C_2 \wedge C_3$



Prefix(φ): all prefixes of φ

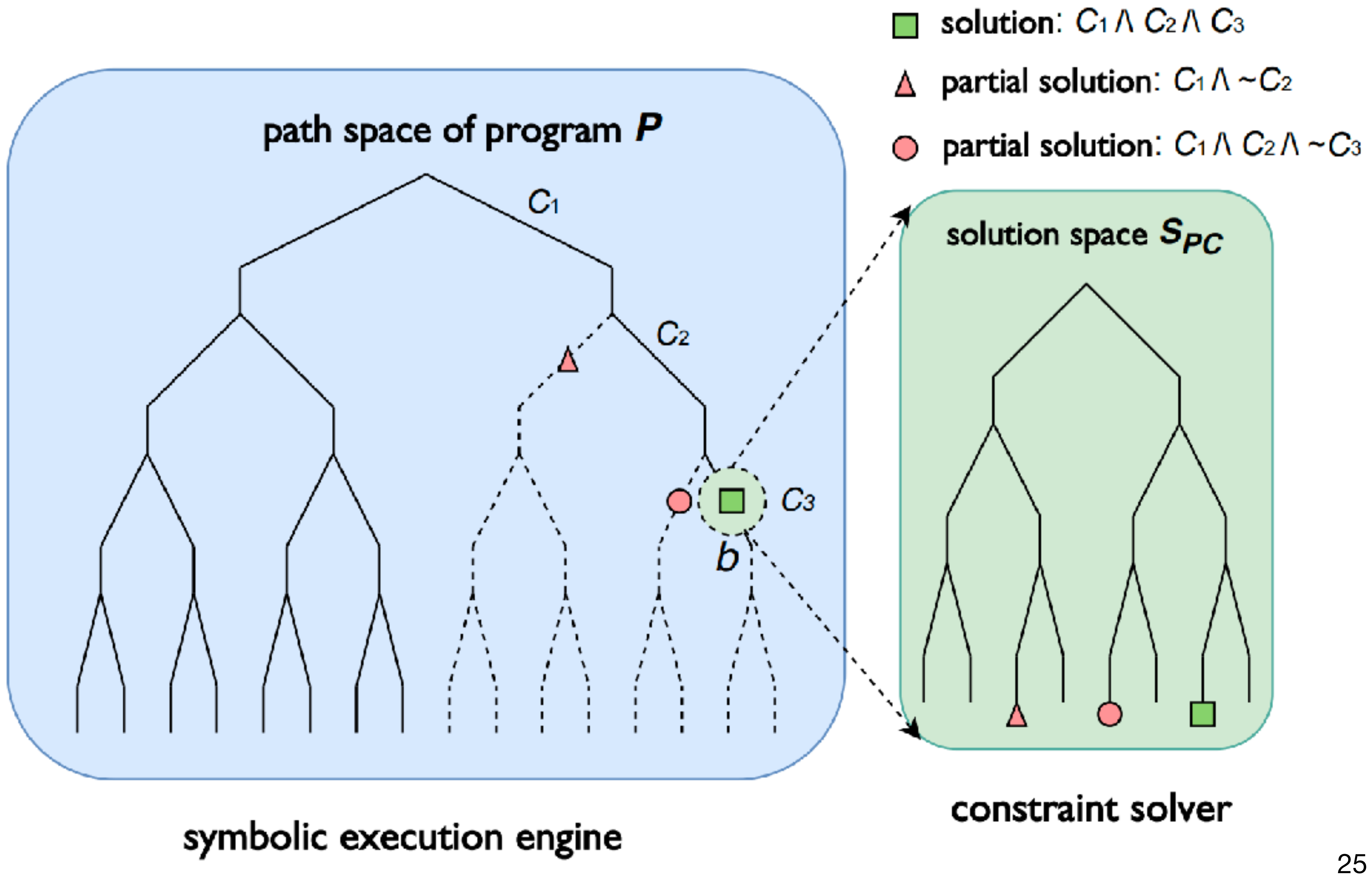
OffThePath(φ): path constraints of φ's off-the-path branches

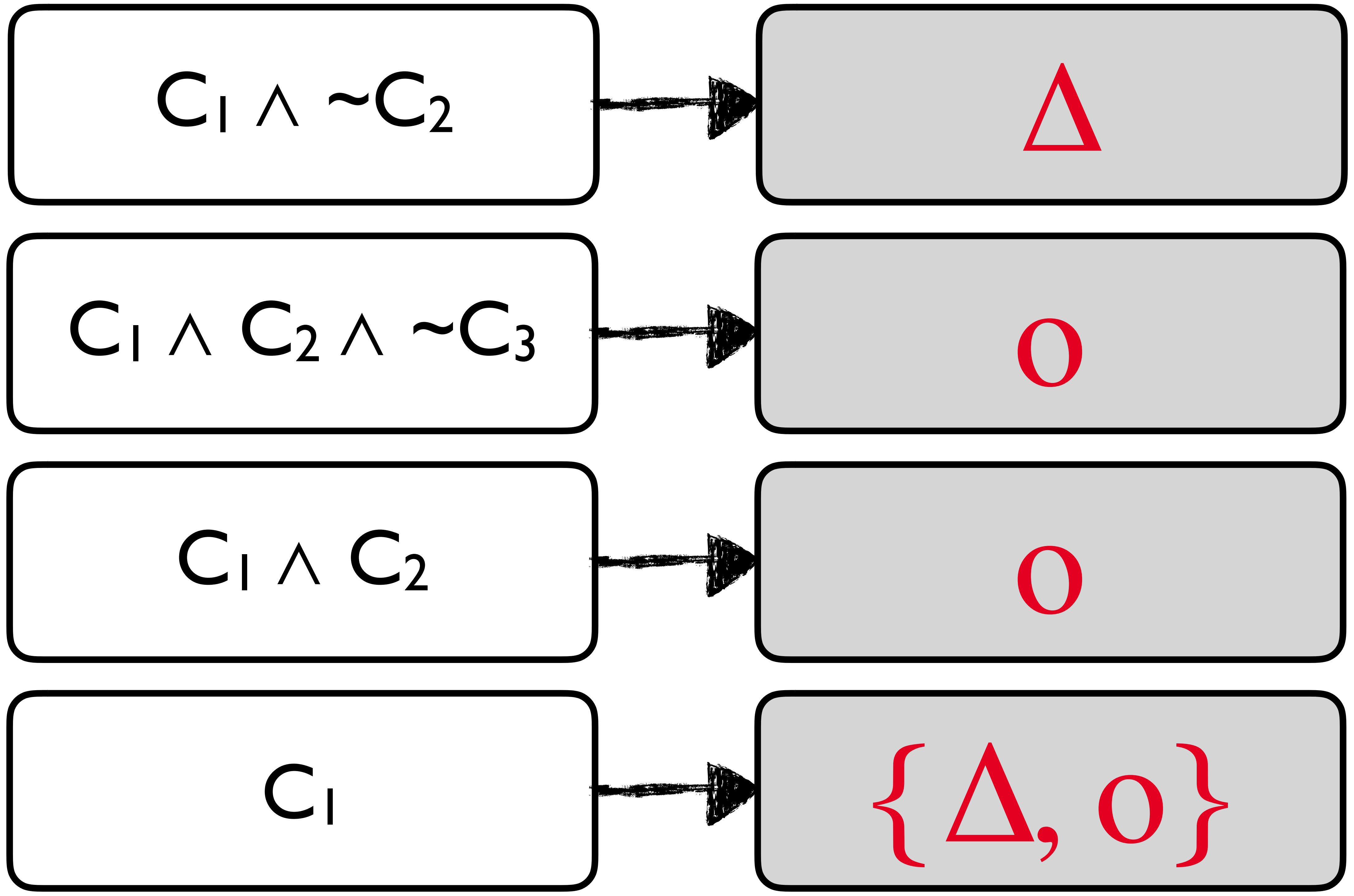
$Prefix(\varphi) = \{C_1, C_1 \wedge C_2\}$

$OffThePath(\varphi) = \{\sim C_1, C_1 \wedge \sim C_2, C_1 \wedge C_2 \wedge \sim C_3\}$

Internals: Expand the Cache

- Construct cache entries: Prefix(φ) $\cup$ OffThePath(φ)
- Extend the single solution of cache entry to a solution set

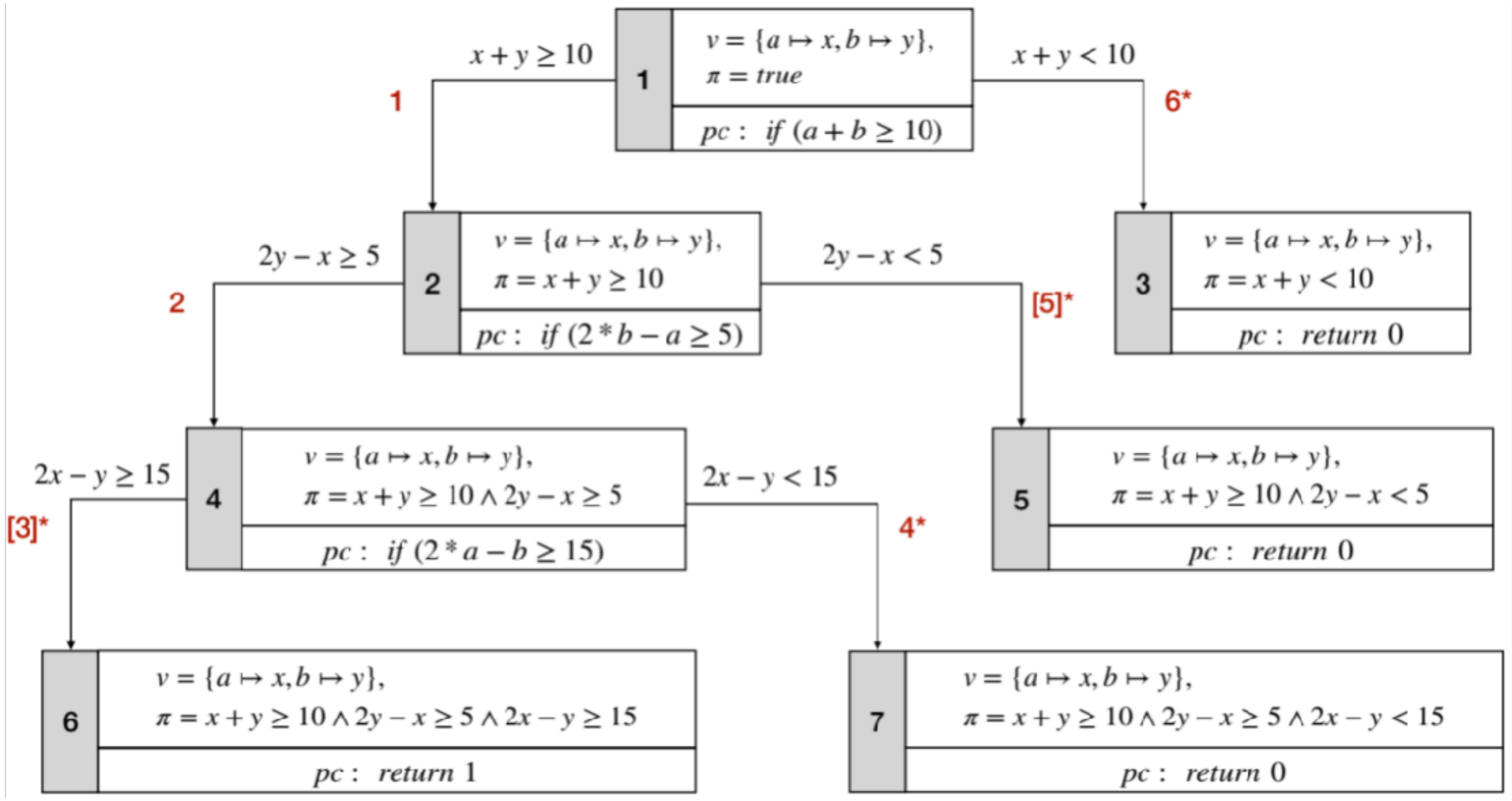




Toy Example

```
int test(int8_t a, int8_t b) {
    if (a + b >= 10) {
        if (2*b - a >= 5) {
            if (2*a - b >= 15) {
                return 1;
            }
        }
    }

    return 0;
}
```



Depth-First Search (DFS) Heuristic

Example: Traditional SE

- SE needs to decide the satisfiability of 6 SMT queries
  - $x+y \geq 10$ with solution $\{x = 10, y = 0\}$
  - $x+y \geq 10 \wedge 2y-x \geq 5$ with solution $\{x = 26, y = 17\}$
  - $x+y \geq 10 \wedge 2y-x \geq 5 \wedge 2x-y \geq 15$ with solution $\{x = 20, y = 20\}$
  - $x+y \geq 10 \wedge 2y-x \geq 5 \wedge 2x-y < 15$ with solution $\{x = 7, y = 31\}$
  - $x+y \geq 10 \wedge 2y-x < 5$ with solution $\{x = 10, y = 1\}$
  - $x+y < 10$ with solution $\{x = 0, y = 9\}$

Need 6 times of solving

Example: Subset-based Caching

- SE needs to decide the satisfiability of 6 SMT queries
  - $x+y \geq 10$ with solution $\{x = 10, y = 0\}$
  - $x+y \geq 10 \wedge 2y-x \geq 5$ with solution $\{x = 26, y = 17\}$
  - $x+y \geq 10 \wedge 2y-x \geq 5 \wedge 2x-y \geq 15$ cached by solution $\{x = 26, y = 17\}$
  - $x+y \geq 10 \wedge 2y-x \geq 5 \wedge 2x-y < 15$ with solution $\{x = 7, y = 31\}$
  - $x+y \geq 10 \wedge 2y-x < 5$ cached by solution $\{x = 10, y = 0\}$
  - $x+y < 10$ with solution $\{x = 0, y = 9\}$

Need 4 times of solving

Example: PS-Based Caching

- SE needs to decide the satisfiability of 6 SMT queries
  - $x+y \geq 10$ with solution $\{x = 10, y = 0\}$
  - $x+y \geq 10 \wedge 2y-x \geq 5$ with solution $\{x = 26, y = 17\}$

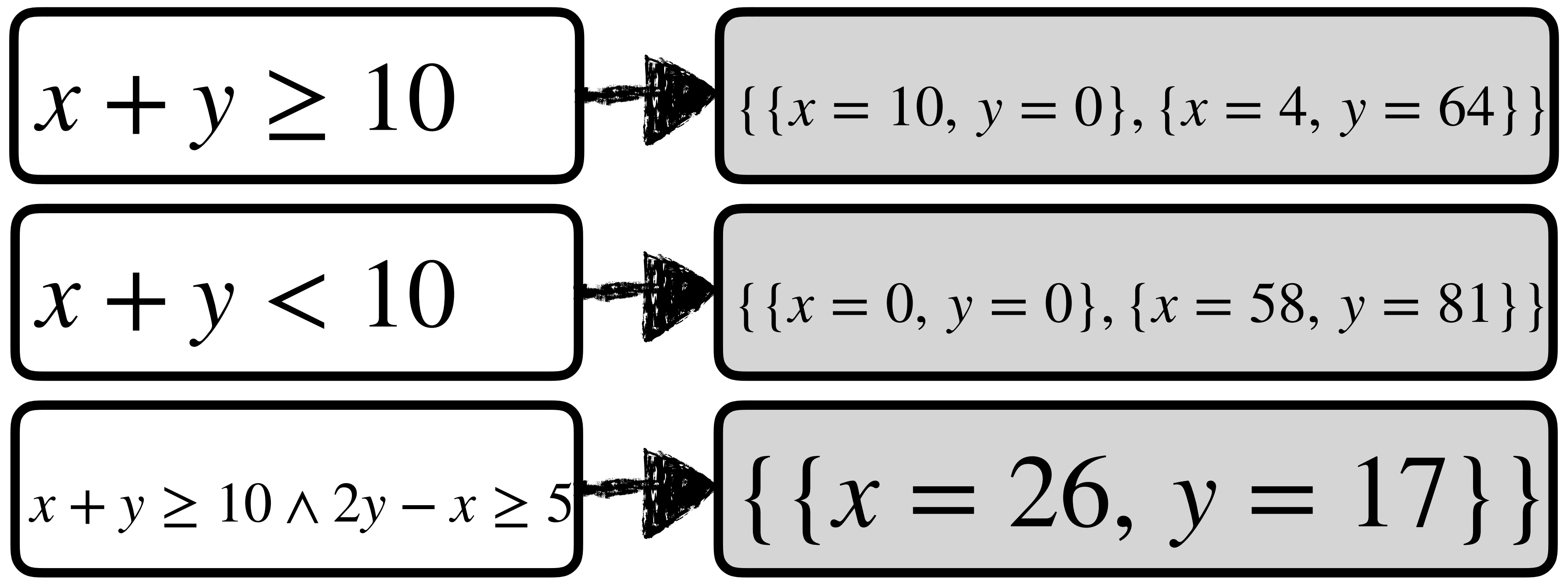
Too simple to produce any ps

Three partial solutions:
$\{x = 58, y = 81\}$,
$\{x = 4, y = 64\}$,
$\{x = 0, y = 0\}$

$x + y \geq 10$ →	$\{\{x = 10, y = 0\}, \{x = 4, y = 64\}\}$
$x + y < 10$ →	$\{\{x = 0, y = 0\}, \{x = 58, y = 81\}\}$
$x + y \geq 10 \wedge 2y - x \geq 5$ →	$\{\{x = 26, y = 17\}\}$

Example: PS-Based Caching

- SE needs to decide the satisfiability of 6 SMT queries
  - $x+y \geq 10$ with solution $\{x = 10, y = 0\}$
  - $x+y \geq 10 \wedge 2y-x \geq 5$ with solution $\{x = 26, y = 17\}$
  - $x+y \geq 10 \wedge 2y-x \geq 5 \wedge 2x-y \geq 15$ cached by solution $\{x = 26, y = 17\}$
  - $x+y \geq 10 \wedge 2y-x \geq 5 \wedge 2x-y < 15$ cached by solution $\{x = 4, y = 64\}$
  - $x+y \geq 10 \wedge 2y-x < 5$ cached by solution $\{x = 10, y = 0\}$
  - $x+y < 10$ cached by solution $\{x = 0, y = 0\}$



Need 2 times of solving

Implementation

- Two types of symbolic executors
  - KLEE / KLEE-Float (C): Subset&Supset-based caching
  - Grulia service in SPF & Green (Java): Utopia[TSE'21]
- Two SMT solvers
  - STP: external SAT solver (Minisat[SAT'03])
  - Z3: self-customized SAT solver

Implementation

effectiveness vs. time & memory overhead

- Implement two parameters to achieve trade-off
  - Kp: limits the number of ps returned by the solver
  - Ks: limits the maximum size of the solution set

Evaluation

- Research questions
  - RQ1: Effectiveness - the number of explored paths or states
  - RQ2: Efficiency - path exploration trend
  - RQ3: The impact of parameter tuning (Kp and Ks)

Experimental Setup

Experiments	Symbolic Executor / Timeout	Constraint Solver / Timeout	Kp + Ks	Search Heuristic
QF_ABV based analysis	KLEE / 30min	STP & Minisat / 30s	250 + 50	Depth-First (DFS), Breadth-First (BFS), Random Cover-New (RCN)
QF_ABVFP based analysis	KLEE-Float / 30min	Z3 / 200s	2500 + 2500	
QF_BV based analysis	SPF & Green / 30min	Z3 / 5s	100 + 100	Depth-First (DFS), Breadth-First (BFS)

Benchmark

- Benchmark
  - QF_ABV: 15 real-world open-source C programs
  - QF_ABVFP: 32 randomly chosen GSL functions
  - QF_BV: 13 Java programs from Green benchmark

Results of Effectiveness (QF_ABV)

Programs	Mode	DFS			BFS			RCN		
		#Paths	CHR	TO(s)	#Paths	CHR	TO(s)	#Paths	CHR	TO(s)
apr	P	**107(33.8%)**	0.847	1.24	84(10.5%)	0.703	0.77	87(29.9%)	0.765	0.87
	O	80	0.793	0.0	76	0.671	0.0	67	0.721	0.0
cmark	P	1790(-7.3%)	0.485	213.49	1569(3.8%)	0.528	28.84	**2569(5.9%)**	0.576	28.01
	O	1931	0.463	0.0	1511	0.502	0.0	2425	0.546	0.0
fribidi	P	336(23.1%)	0.924	9.75	**8427(24.3%)**	0.975	6.09	6925(2.2%)	0.971	4.19
	O	273	0.909	0.0	6782	0.966	0.0	6773	0.966	0.0
gas	P	312(2.3%)	0.888	6.03	**1541(12.2%)**	0.895	1.09	2563(-13.3%)	0.906	0.9
	O	305	0.876	0.0	1373	0.87	0.0	2955	0.906	0.0
json-c	P	409(3.0%)	0.565	65.39	569(28.7%)	0.557	71.71	**492(41.8%)**	0.556	67.1
	O	397	0.35	0.0	442	0.236	0.0	347	0.209	0.0
libinjection	P	1007(-15.7%)	0.936	1.87	**27373(1.7%)**	0.936	4.3	16088(1.5%)	0.928	2.77
	O	1194	0.946	0.0	26918	0.939	0.0	15856	0.924	0.0
libtommath	P	38006(5.1%)	0.933	24.62	27214(33.7%)	0.923	8.36	**42238(98.7%)**	0.96	6.63
	O	36151	0.915	0.0	20347	0.917	0.0	21260	0.964	0.0
m4	P	14938(-26.4%)	0.965	124.29	75509(1.2%)	0.996	131.61	**44140(31.4%)**	0.99	129.03
	O	20304	0.969	0.0	74645	0.995	0.0	33586	0.983	0.0
discount	P	**86474(24.0%)**	0.974	25.74	222984(9.8%)	0.999	3.68	274828(4.3%)	0.998	2.62
	O	69764	0.972	0.0	203125	0.998	0.0	263373	0.997	0.0
pacparser	P	36(2.9%)	0.663	12.02	3668(5.9%)	0.96	5.13	**5655(9.9%)**	0.971	6.95
	O	35	0.641	0.0	3465	0.95	0.0	5145	0.96	0.0
ptx	P	**3312(28.4%)**	0.985	10.67	151(-5.0%)	0.696	63.74	446(-6.3%)	0.778	29.74
	O	2579	0.982	0.0	159	0.682	0.0	476	0.779	0.0
sha1-cd	P	80(9.6%)	0.596	38.26	**1772(68.0%)**	0.591	266.96	3180(34.3%)	0.568	551.15
	O	73	0.52	0.0	1055	0.525	0.0	2367	0.52	0.0
smaz	P	52(26.8%)	0.622	34.65	154(51.0%)	0.628	17.81	**106(89.3%)**	0.619	19.84
	O	41	0.601	0.0	102	0.564	0.0	56	0.58	0.0
sqlite3	P	42(-10.6%)	0.85	8.68	225(-6.2%)	0.823	20.96	**173(8.8%)**	0.685	11.95
	O	47	0.839	0.0	240	0.801	0.0	159	0.581	0.0
sundown	P	1524(6.6%)	0.792	74.05	16244(0.0%)	0.84	50.07	**26906(7.9%)**	0.883	46.97
	O	1430	0.723	0.0	16241	0.831	0.0	24929	0.87	0.0

Search Heuristic	Improvement (%)
DFS	7.0
BFS	16.0
RCN	23.1

Improve the numbers of paths for 9 programs under all search heuristics

Results of Effectiveness (QF_ABVFP)

GSL Functions	Mode	DFS			BFS			RCN		
		#Paths	CHR	TO(s)	#Paths	CHR	TO(s)	#Paths	CHR	TO(s)
gsl_cdf_beta_P	P	19(0.0%)	0.506	5.3	**44(37.5%)**	0.617	3.4	63(21.2%)	0.605	5.9
	O	19	0.455	0.0	32	0.465	0.0	52	0.466	0.0
gsl_cdf_beta_Pinv	P	24(0.0%)	0.566	3.2	40(0.0%)	0.661	2.4	**64(30.6%)**	0.669	4.7
	O	24	0.458	0.0	40	0.503	0.0	49	0.479	0.0
gsl_cdf_cauchy_Pinv	P	17(21.4%)	0.565	3.9	44(10.0%)	0.63	5.0	**47(27.0%)**	0.663	6.1
	O	14	0.517	0.0	40	0.469	0.0	37	0.466	0.0
gsl_cdf_chisq_P	P	15(7.1%)	0.581	1.8	**39(25.8%)**	0.661	3.8	43(22.9%)	0.643	4.1
	O	14	0.478	0.0	31	0.5	0.0	35	0.49	0.0
gsl_cdf_gumbel1_Q	P	14(16.7%)	0.571	1.1	**27(42.1%)**	0.634	1.3	33(17.9%)	0.602	1.8
	O	12	0.462	0.0	19	0.573	0.0	28	0.538	0.0
gsl_cdf_gumbel1_Qinv	P	**16(23.1%)**	0.552	1.2	30(11.1%)	0.604	0.7	40(11.1%)	0.614	1.7
	O	13	0.471	0.0	27	0.474	0.0	36	0.476	0.0
gsl_cdf_gumbel2_Pinv	P	24(26.3%)	0.588	0.8	45(7.1%)	0.651	1.5	**64(45.5%)**	0.675	1.5
	O	19	0.519	0.0	42	0.558	0.0	44	0.507	0.0
gsl_cdf_weibull_Qinv	P	23(9.5%)	0.608	1.1	40(0.0%)	0.629	1.9	**44(10.0%)**	0.639	1.4
	O	21	0.522	0.0	40	0.562	0.0	40	0.552	0.0
gsl_complex_exp	P	**27(58.8%)**	0.604	1.7	238(11.7%)	0.841	2.5	310(13.1%)	0.895	3.0
	O	17	0.516	0.0	213	0.794	0.0	274	0.848	0.0
gsl_complex_log	P	**27(58.8%)**	0.603	1.5	238(11.7%)	0.845	2.6	295(4.6%)	0.892	3.1
	O	17	0.513	0.0	213	0.794	0.0	282	0.85	0.0
gsl_complex_sinh	P	29(45.0%)	0.604	1.5	143(5.1%)	0.721	5.4	**284(68.0%)**	0.759	8.2
	O	20	0.516	0.0	136	0.681	0.0	169	0.633	0.0
gsl_deriv_forward	P	11(22.2%)	0.589	1.6	**13(30.0%)**	0.625	1.9	15(25.0%)	0.647	1.6
	O	9	0.462	0.0	10	0.459	0.0	12	0.457	0.0
gsl_diff_forward	P	10(11.1%)	0.541	0.2	10(42.9%)	0.57	0.7	**15(50.0%)**	0.604	0.6
	O	9	0.513	0.0	7	0.506	0.0	10	0.493	0.0
gsl_eigen_genv_sort	P	17(30.8%)	0.617	1.4	75(27.1%)	0.833	2.9	**50(56.2%)**	0.656	2.9
	O	13	0.459	0.0	59	0.783	0.0	32	0.519	0.0
gsl_integration_glfixed	P	11(10.0%)	0.569	3.8	22(15.8%)	0.628	3.2	**29(20.8%)**	0.582	4.3
	O	10	0.477	0.0	19	0.493	0.0	24	0.5	0.0
gsl_integration_qawc	P	138(137.9%)	0.776	195.2	24(0.0%)	0.614	1.0	**243(170.0%)**	0.796	13.1
	O	58	0.489	0.0	24	0.557	0.0	90	0.628	0.0
gsl_integration_qng	P	12(9.1%)	0.579	3.2	**70(311.8%)**	0.679	2.8	109(22.5%)	0.739	6.8
	O	11	0.469	0.0	17	0.509	0.0	89	0.619	0.0
gsl_linalg_PTLQ_decomp	P	44(18.9%)	0.538	7.0	**113(41.2%)**	0.553	2.1	34(13.3%)	0.508	2.1
	O	37	0.494	0.0	80	0.52	0.0	30	0.444	0.0
gsl_linalg_complex_LU_lndet	P	**30(57.9%)**	0.726	6.7	79(14.5%)	0.731	2.3	47(-2.1%)	0.627	2.8
	O	19	0.476	0.0	69	0.543	0.0	48	0.491	0.0
gsl_poly_complex_solve_cubic	P	23(1050.0%)	0.776	3.0	25(1150.0%)	0.74	2.1	**28(1300.0%)**	0.753	2.5
	O	2	0.286	0.0	2	0.286	0.0	2	0.286	0.0
gsl_poly_complex_solve_quadratic	P	20(66.7%)	0.746	4.9	**60(160.9%)**	0.676	5.8	58(123.1%)	0.697	5.6
	O	12	0.459	0.0	23	0.482	0.0	26	0.484	0.0
gsl_poly_solve_quadratic	P	19(46.2%)	0.719	5.1	**26(62.5%)**	0.65	2.7	27(50.0%)	0.66	2.9
	O	13	0.463	0.0	16	0.475	0.0	18	0.477	0.0
gsl_sf_airy_Ai_deriv_e	P	92(0.0%)	0.623	26.8	13(0.0%)	0.516	0.1	**178(85.4%)**	0.827	7.5
	O	92	0.514	0.0	13	0.516	0.0	96	0.5	0.0
gsl_sf_bessel_In_scaled_e	P	**95(61.0%)**	0.717	30.2	58(28.9%)	0.559	4.9	48(11.6%)	0.614	2.3
	O	59	0.481	0.0	45	0.5	0.0	43	0.51	0.0
gsl_sf_bessel_Inu_e	P	12(0.0%)	0.526	1.4	**37(60.9%)**	0.633	4.0	35(9.4%)	0.616	4.4
	O	12	0.459	0.0	23	0.496	0.0	32	0.497	0.0
gsl_sf_bessel_Inu_scaled_asymp_unif_e	P	**27(107.7%)**	0.762	5.7	11(10.0%)	0.647	1.1	18(63.6%)	0.697	1.4
	O	13	0.471	0.0	10	0.468	0.0	11	0.465	0.0
gsl_sf_bessel_Inu_scaled_e	P	12(9.1%)	0.538	1.4	**36(50.0%)**	0.624	4.3	37(27.6%)	0.638	4.6
	O	11	0.457	0.0	24	0.496	0.0	29	0.496	0.0
gsl_sf_bessel_cos_pi4_e	P	**25(92.3%)**	0.573	1.2	106(3.9%)	0.685	4.0	139(37.6%)	0.744	5.0
	O	13	0.515	0.0	102	0.636	0.0	101	0.582	0.0
gsl_sf_bessel_sin_pi4_e	P	**23(64.3%)**	0.572	1.2	106(2.9%)	0.687	3.8	139(44.8%)	0.744	5.0
	O	14	0.511	0.0	103	0.634	0.0	96	0.581	0.0
gsl_sf_debye_1_e	P	165(217.3%)	0.827	32.5	79(68.1%)	0.78	11.4	**144(585.7%)**	0.831	28.6
	O	52	0.492	0.0	47	0.492	0.0	21	0.491	0.0
gsl_sf_elljac_e	P	94(-1.1%)	0.605	88.6	71(9.2%)	0.584	3.3	**108(16.1%)**	0.634	5.6
	O	95	0.497	0.0	65	0.502	0.0	93	0.493	0.0
gsl_sf_gamma_inc_P_e	P	14(-6.7%)	0.49	1.9	49(14.0%)	0.653	4.8	**47(20.5%)**	0.665	5.6
	O	15	0.468	0.0	43	0.512	0.0	39	0.497	0.0

Search Heuristic	Improvement (%)
DFS	71.0
BFS	70.8
RCN	93.8

Improve the numbers of paths for 29 functions under all search heuristics

Results of Effectiveness (QF_ABVFP)

- The ratios of UNSAT queries among those reaching the solver vary a lot
  - QF_ABV: DFS (0.7), BFS (0.6), RCN (0.61)
  - QF_ABVFP: DFS (0.4), BFS (0.47), RCN (0.4)

Search Heuristic	Improvement (%)
DFS	71.0
BFS	70.8
RCN	93.8

The PS-based caching mechanism can only handle SAT queries

Results of Effectiveness (QF_BV)

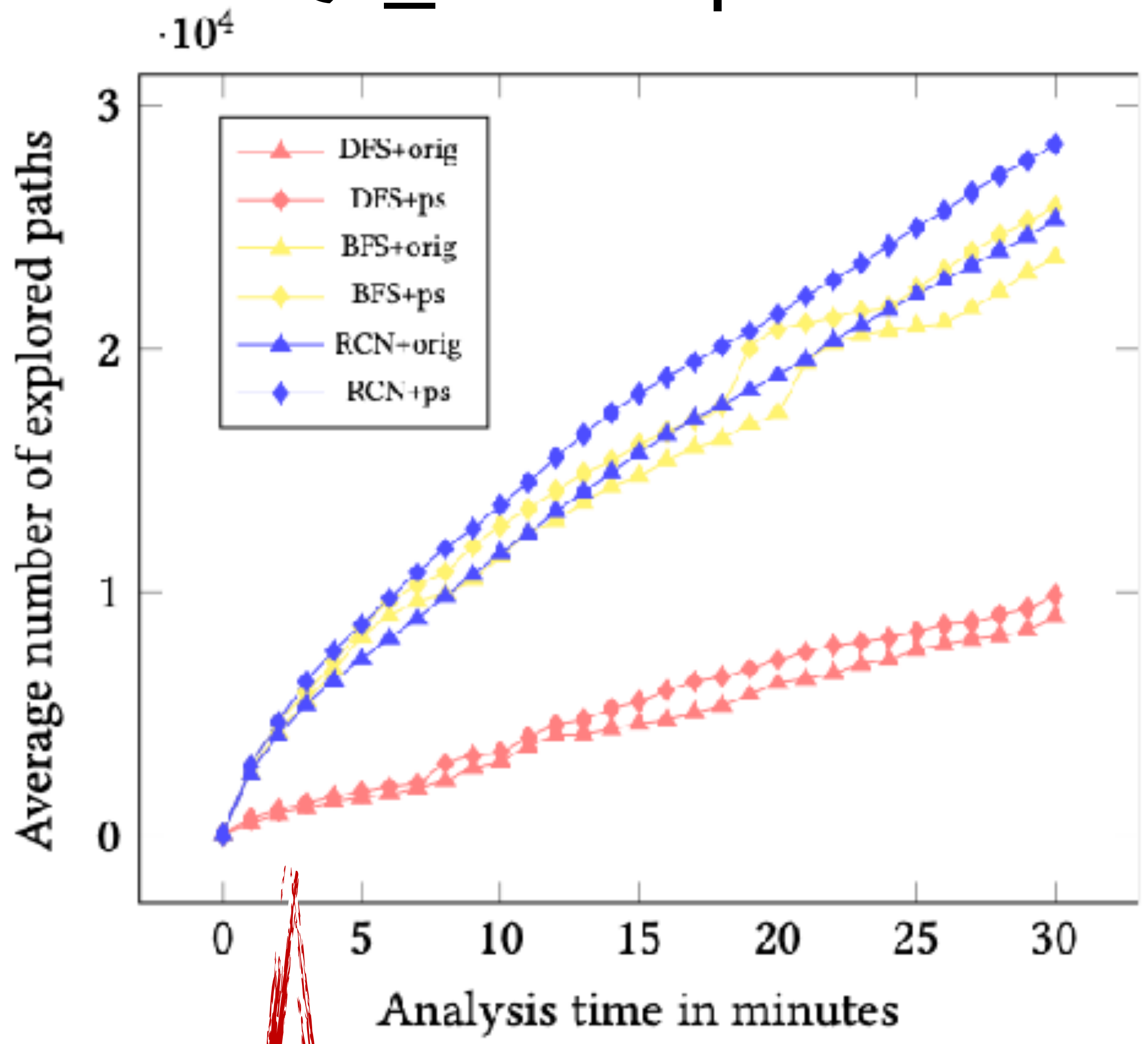
Programs	Mode	DFS				BFS			
		#States	CHR	TO(s)	T(s)	#States	CHR	TO(s)	T(s)
Remainder	P	957(0.0%)	0.761	160.6	**285.8**	957(0.0%)	0.771	178.6	298.6
	O	957	0.24	0.0	491.0	957	0.156	0.0	551.6
BubbleSort	P	11777(17.9%)	0.283	312.5	1802.0	**18255(41.8%)**	0.323	159.7	1800.2
	O	9992	0.044	0.0	1800.0	12876	0.144	0.0	1800.0
Dijkstra	P	8242(60.5%)	0.325	113.1	1800.0	**8294(63.5%)**	0.347	105.3	1800.0
	O	5135	0.116	0.0	1800.0	5072	0.125	0.0	1800.0
NanoXML	P	**48520(227.3%)**	0.66	6.7	1800.0	99426(51.6%)	0.536	9.1	1800.0
	O	14822	0.362	0.0	1800.0	65598	0.533	0.0	1800.0
SortedListInt	P	**79704(377.2%)**	0.846	57.8	1800.0	38491(95.5%)	0.817	11.8	1800.0
	O	16703	0.686	0.0	1800.0	19684	0.695	0.0	1800.0
BinTree	P	**15227(47.1%)**	0.911	14.8	**912.6**	15227(21.6%)	0.899	17.2	1204.4
	O	10353	0.829	0.0	1800.0	12522	0.842	0.0	1800.0
Triangle	P	2207(0.0%)	0.502	1.0	**7.2**	2207(0.0%)	0.474	1.5	19.8
	O	2207	0.354	0.0	90.0	2207	0.303	0.0	111.4
Operations	P	15619(0.0%)	0.52	59.7	**256.4**	15619(0.0%)	0.465	58.7	262.8
	O	15619	0.188	0.0	544.2	15619	0.222	0.0	591.2
Sorting	P	**24219(80.0%)**	0.561	127.3	1801.0	18714(25.1%)	0.36	48.6	1800.0
	O	13459	0.256	0.0	1800.0	14959	0.181	0.0	1800.0
MagicIndex	P	**7202(52.4%)**	0.944	0.2	**9.2**	7202(0.0%)	0.956	0.2	10.8
	O	4726	0.551	0.0	1788.6	7202	0.786	0.0	871.2
BinomialHeap	P	**47461(106.5%)**	0.939	86.7	**1435.4**	47303(21.1%)	0.939	65.6	1493.2
	O	22978	0.885	0.0	1800.0	39063	0.925	0.0	1800.0
TreeMap	P	24640(120.2%)	0.942	58.7	1800.0	**61411(131.7%)**	0.961	30.1	1800.0
	O	11188	0.888	0.0	1800.0	26499	0.93	0.0	1800.0
Median	P	12452(53.5%)	0.29	313.7	1802.0	**17504(59.2%)**	0.325	156.9	1800.0
	O	8111	0.046	0.0	1800.0	10994	0.14	0.0	1800.0

Search Heuristic	**Improvement (%)**
DFS	114.3
BFS	56.8

- The speedups of early completed tasks
  - DFS: 1.72x ~ 12.5x
  - BFS: 1.85x ~ 80.67x

Results of Efficiency

QF_ABV Experiment



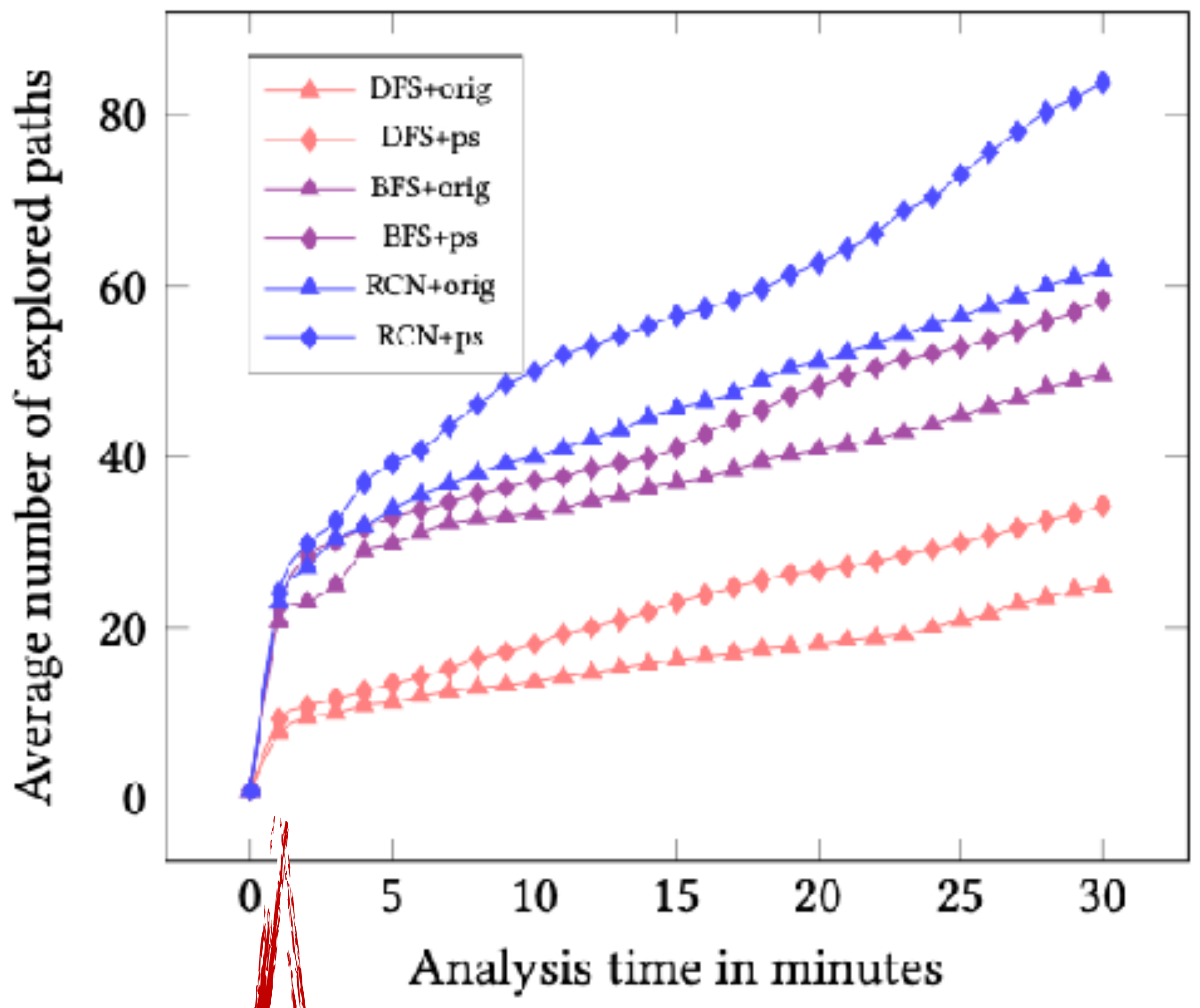
Speedup

DFS: 1.07x

BFS: 1.11x

RCN: 1.15x

QF_ABVFP Experiment



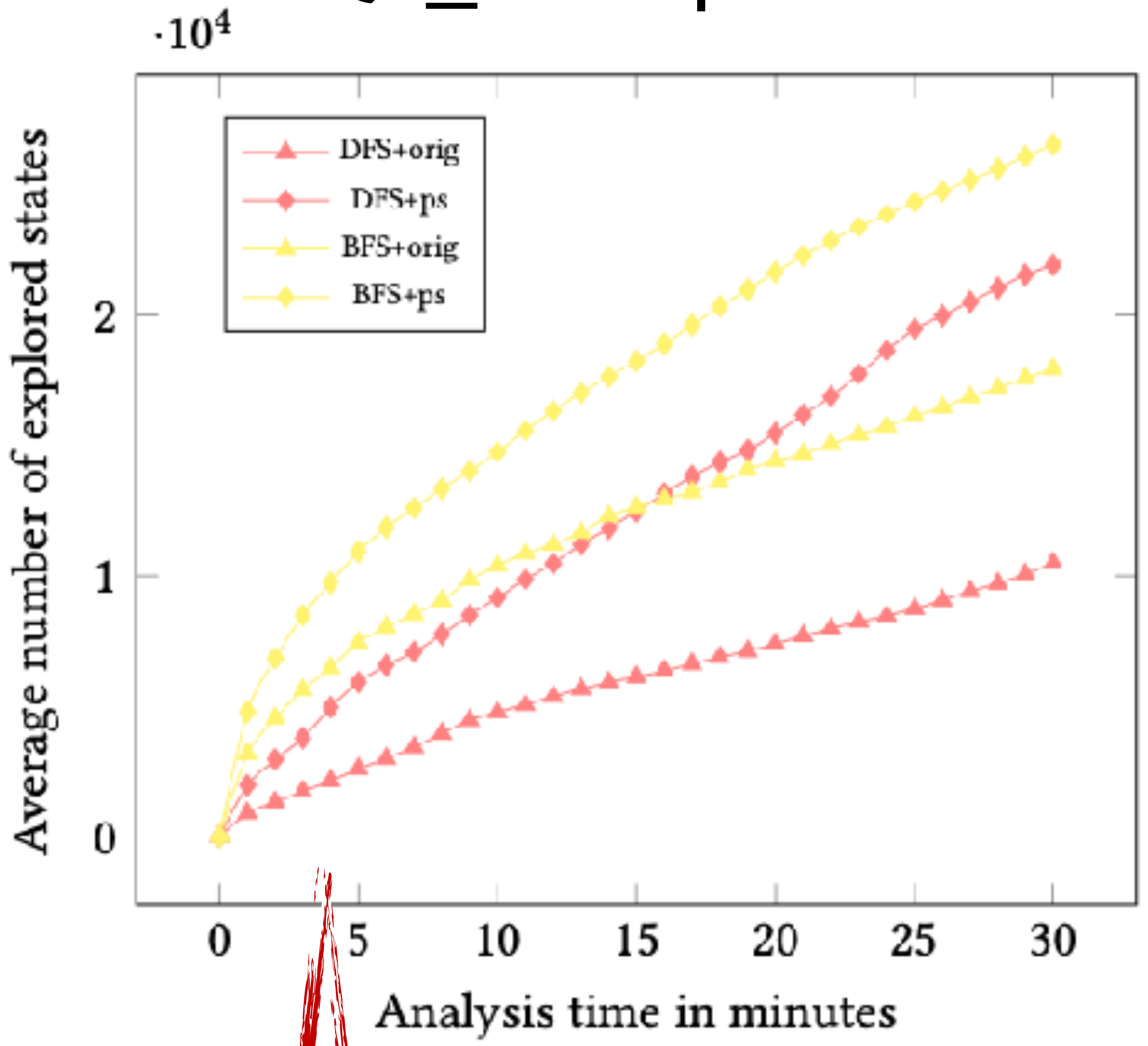
Speedup

DFS: 1.67x

BFS: 1.36x

RCN: 1.50x

QF_BV Experiment

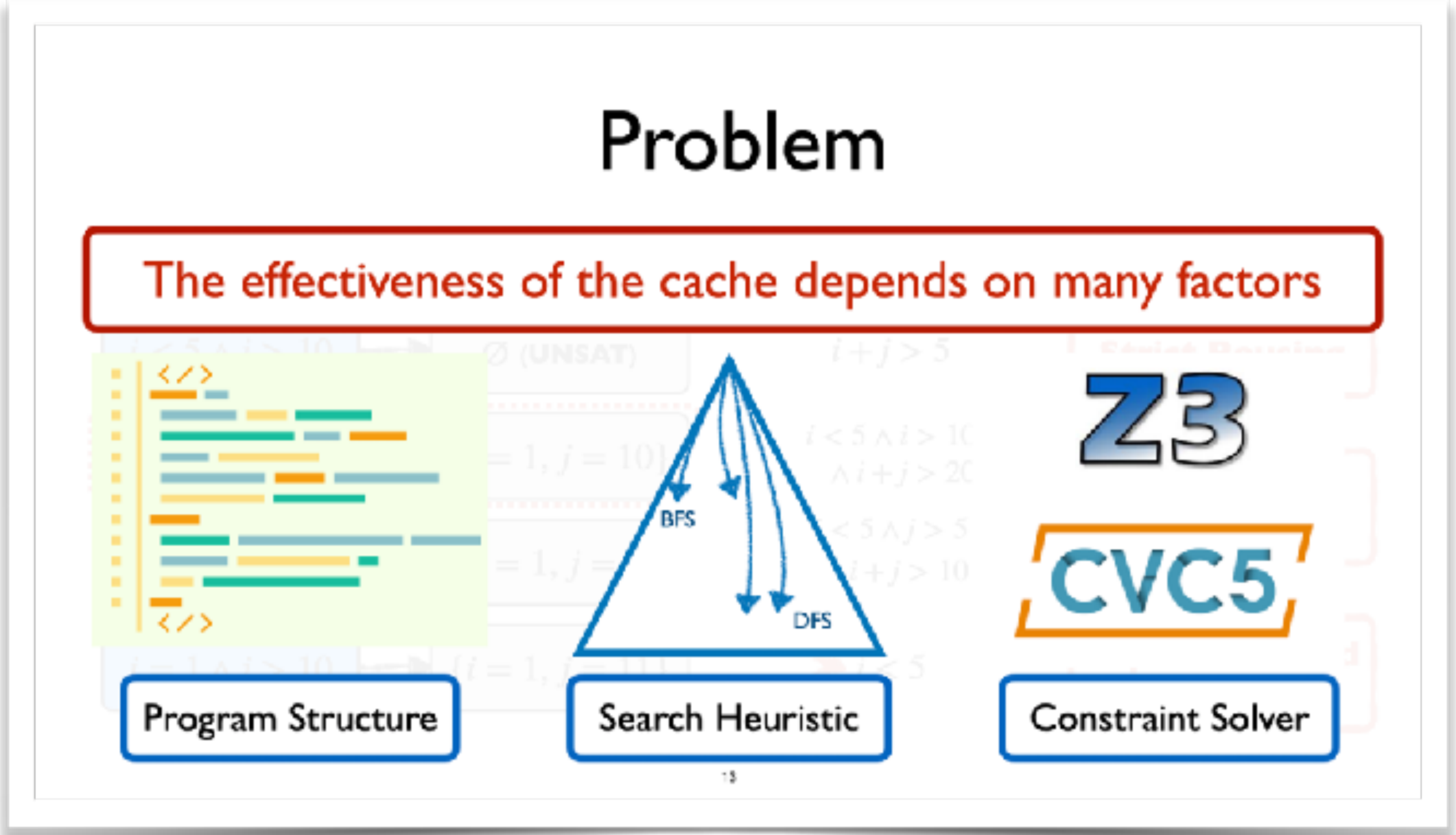


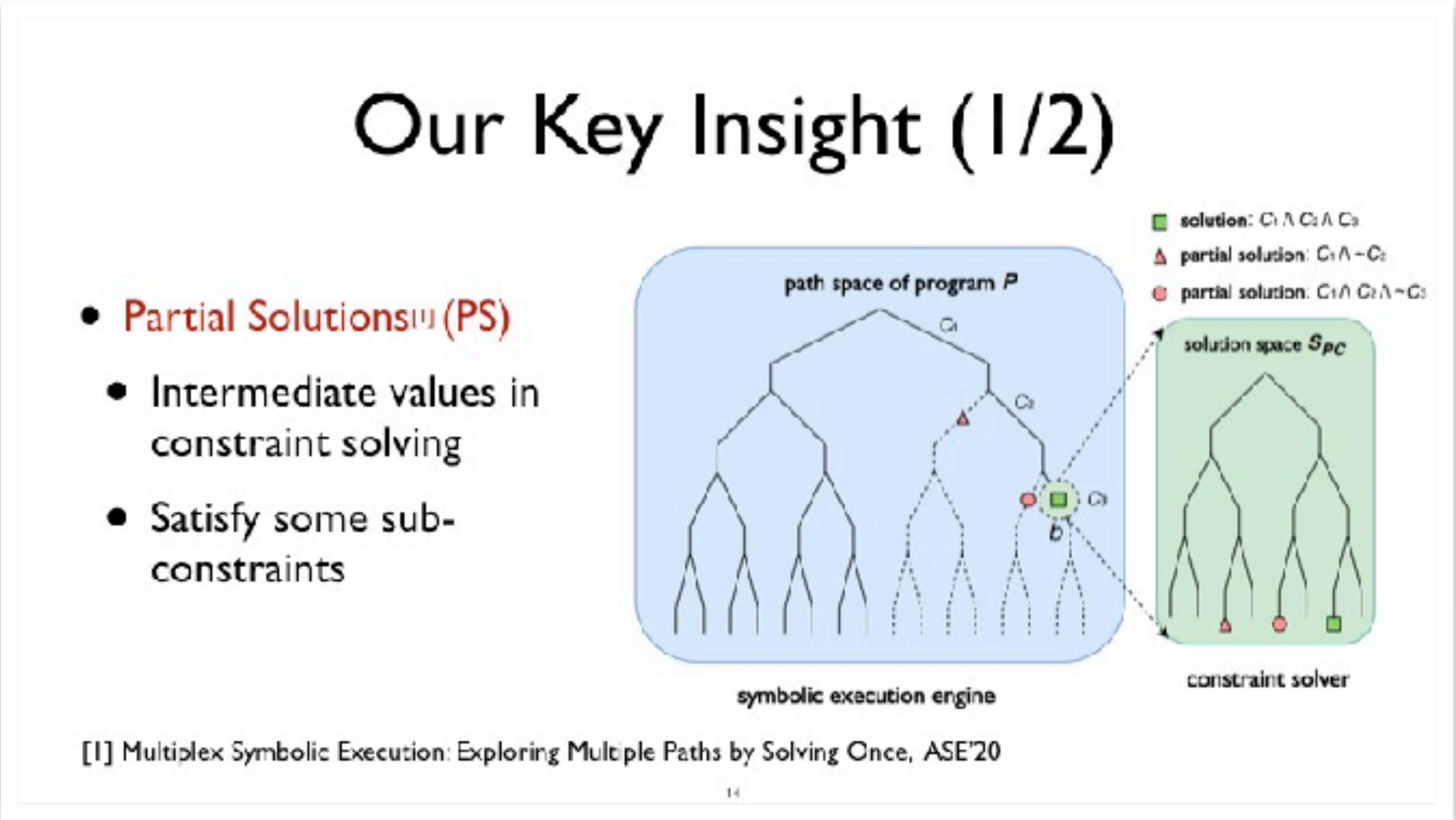
Speedup

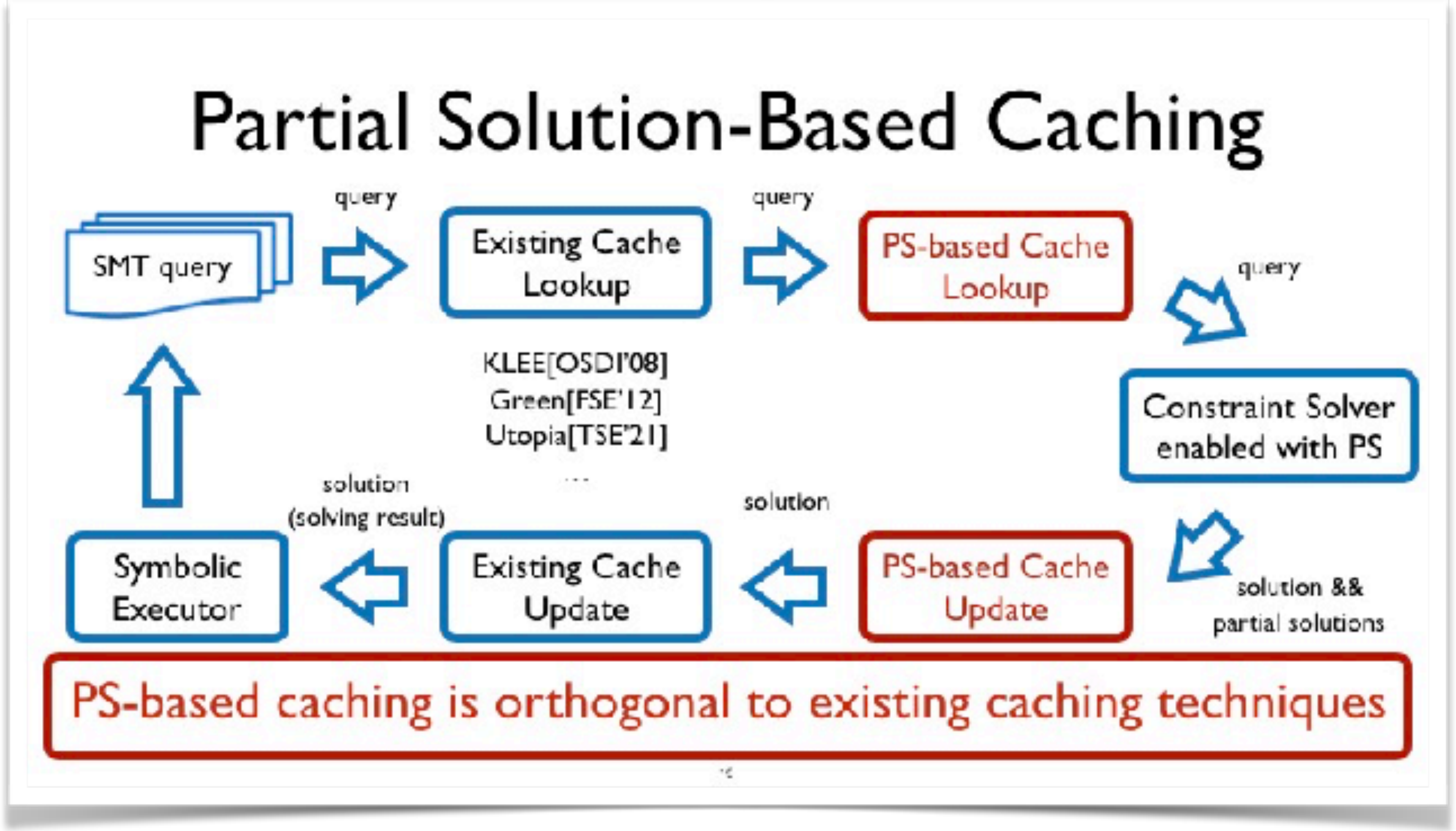
DFS: 2.3x

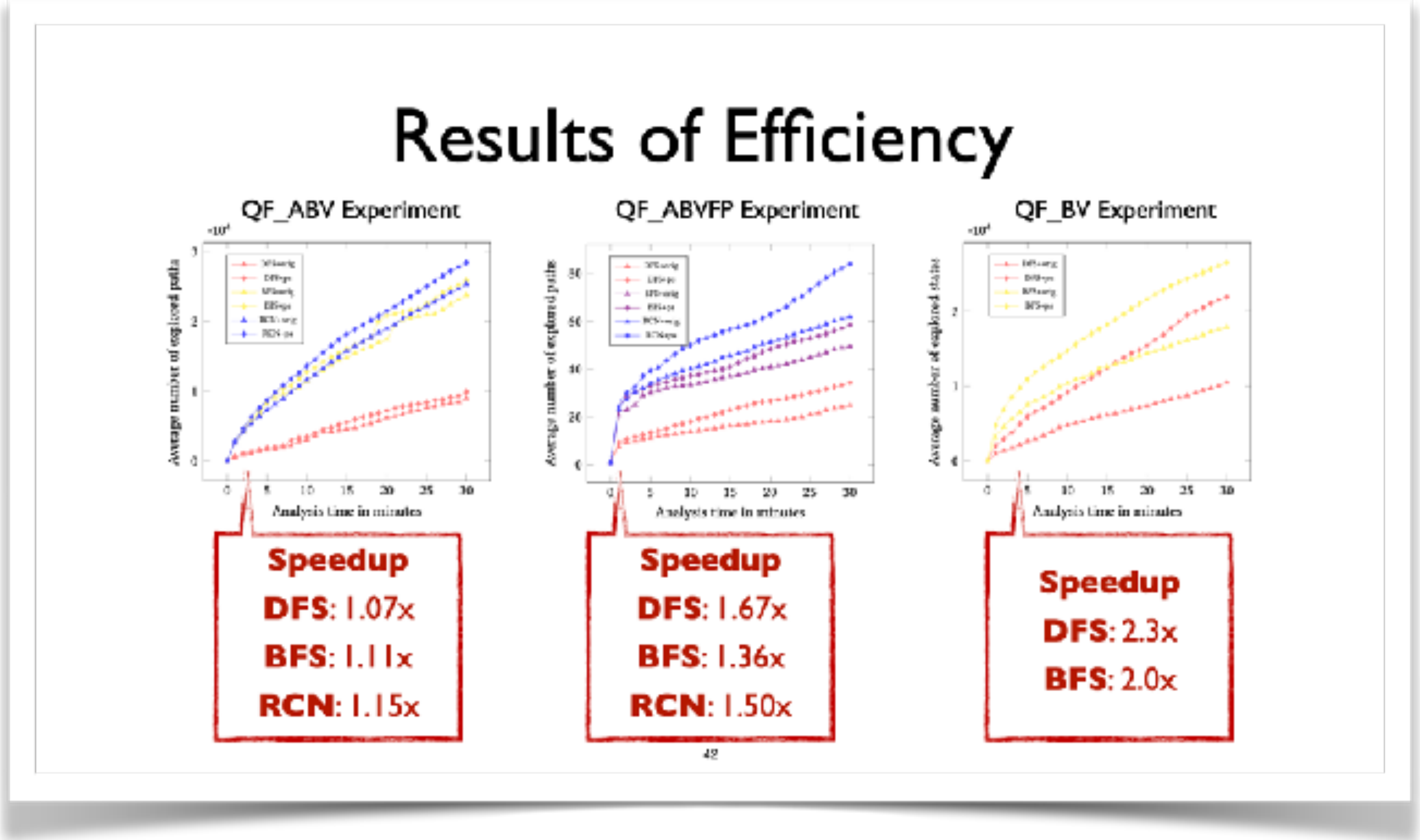
BFS: 2.0x

Conclusion


Problem
The effectiveness of the cache depends on many factors
BFS
DFS
Z3
CVC5
Program Structure
Search Heuristic
Constraint Solver


Our Key Insight (1/2)
• Partial Solutions[1] (PS)
• Intermediate values in constraint solving
• Satisfy some sub-constraints
path space of program P
solution space S_{PC}
symbolic execution engine
constraint solver
[1] Multiplex Symbolic Execution: Exploring Multiple Paths by Solving Once, ASE'20


Partial Solution-Based Caching
SMT query
query
Existing Cache Lookup
query
PS-based Cache Lookup
query
Constraint Solver enabled with PS
KLEE[OSDI'08]
Green[FSE'12]
Utopia[TSE'21]
...
solution (solving result)
solution
solution & partial solutions
Symbolic Executor
Existing Cache Update
PS-based Cache Update
PS-based caching is orthogonal to existing caching techniques


Results of Efficiency
QF_ABV Experiment
QF_ABVFP Experiment
QF_BV Experiment
Speedup
DFS: 1.07x
BFS: 1.11x
RCN: 1.15x
Speedup
DFS: 1.67x
BFS: 1.36x
RCN: 1.50x
Speedup
DFS: 2.3x
BFS: 2.0x

Thank you!

Q&A

Artifact

https://github.com/zbchen/pscache