Extension of the EZSMT System for Non-tight Programs

Yuliya Lierler and Da Shen, University of Nebraska at Omaha

Supported by the 2017 Fund for Undergraduate Scholarly Experiences Grant

Background Technology

Answer Set Programming (ASP):

- A computer programming language in artificial intelligence
- Users state specifications, called programs, for tasks
- No need to worry about how solutions are computed
- Plays a critical role in development of software in science, humanities, and industry
- Has issues when possible solutions grow quickly over time

Constraint Answer Set Programming (CASP):

- An integration of ASP and constraint processing
- Tackles several issues of ASP
- Solvers such as CLINGCON

Satisfiability Modulo Theories (SMT) solvers:

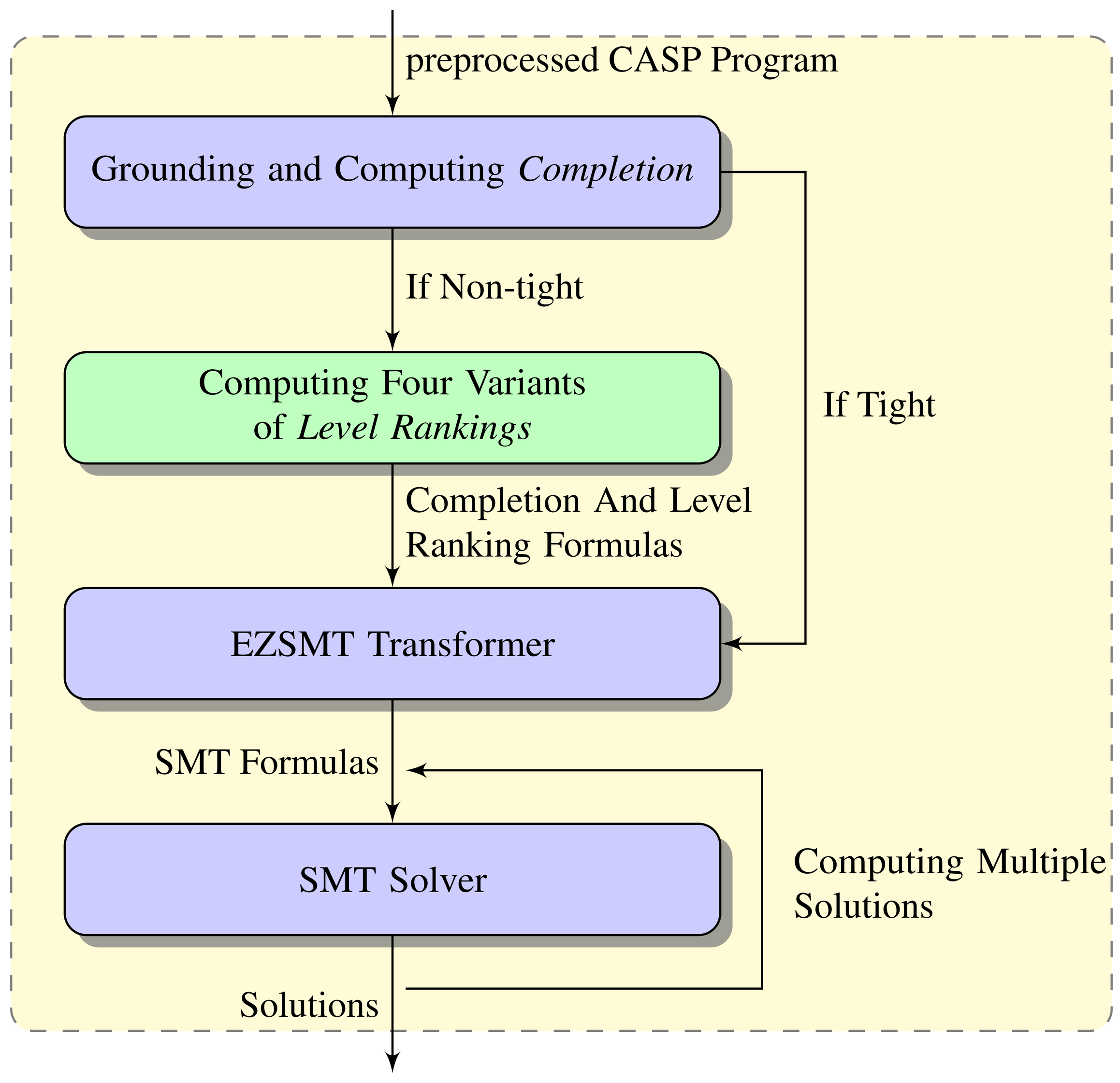
- High-performance tools stemming from software verification community

The EZSMT System

- A software system in artificial intelligence
- Automatically finds solutions to CASP problems
- Utilizes SMT solvers for computation
- Often outperforms its peers
- Unable to process a category of important relations called *non-tight*
  e.g. reachability relations between cities on a map shown in the example marked by blue color

The EZSMT$^+$ System

- We extend EZSMT so that it can take non-tight input. We call the new system EZSMT$^+$.
- EZSMT$^+$ architecture:



- Blue blocks: existing EZSMT system
- Green block: our extension to tackle non-tight input
- EZSMT$^+$ is able to find multiple solutions, which the original EZSMT is unable to do.

Example: Traveling Salesman Problem

Problem Description We are given a map consisting of cities and roads. Each road directly connects a pair of cities, and cost the salesman some time to go through. The salesman can pass each city only once. We are asked to find a route for the salesman to visit all the cities before a given deadline.

Encodings Three approaches: ASP, traditional CASP (CLINGCON) and EZSMT$^+$

Approach	Line	Encoding	Meaning
ASP	1	1 { route(X,Y) : road(X,Y), route(X,Y) : road(Y,X) } 1 :- city(X).	For each city, we choose one route leaving the city.
	2	1 { route(X,Y) : road(X,Y), route(X,Y) : road(Y,X) } 1 :- city(Y).	For each city, we choose one route going to the city.
	3	reached(X) :- initial(X).	The initial city is reached.
	4	reached(Y) :- reached(X), route(X,Y).	If city X is reached and the route from city X to city Y is chosen, then city Y is also reached.
	5	:- city(X), not reached(X).	No city can be not reached.
	6	:- W+1 [route(X,Y) : cost(X,Y,C) = C], maxCost(W).	The total time cost must be less than maximal value.
CLINGCON	7	the same as line 1-5	Go though all cities once.
	8	&sum c(X,Y) = 0 :- cost(X,Y,C), not route(X,Y).	Time spent on a road is 0 if the road is not in our rout
	9	&sum c(X,Y) = C :- cost(X,Y,C), route(X,Y).	Time spent on a road is the cost if the road is in our rou
	10	:- &sum c(X,Y) : cost(X,Y,C) > W, maxCost(W).	The total time cost must be less than maximal value.
EZSMT$^+$	11	the same as line 1-5	Go though all cities once.
	12	cspvar(c(X,Y),0,C) :- cost(X,Y,C).	Declaration of constraint variables.
	13	required(c(X,Y) == 0) :- cost(X,Y,C), not route(X,Y).	Time spent on a road is 0 if the road is not in our rout
	14	required(c(X,Y) == C) :- cost(X,Y,C), route(X,Y).	Time spent on a road is the cost if the road is in our rou
	15	:- required(sum([c/2], >, W)), maxCost(W).	The total time cost must be less than maximal value.

- Green lines: linear constraints, where ASP solvers have issues when a large amount of possible results exist
- Blue lines: non-tight relations, which the original EZSMT system can not handle

Experimental Data

Benchmark	CLINGCON	EZSMT$^+$
RoutingMin(100)	**4.68**	31.2
RoutingMax(100)	3144	**2989**
Trav. Sals.(30)	**455**	3742
Labyrinth*(22)	**3002**(1)	5665(2)

Conclusion

- Pure ASP programs: solved by ASP solvers or SAT solvers
- CASP programs: traditionally solved by ASP solvers and finite domain constraint solvers;
  in EZSMT$^+$ solved by SMT solvers, which are equivalent to SAT solvers and integer linear constraint solvers
- Experimental analysis shows that EZSMT$^+$ is a viable tool for finding solutions to CASP programs.
- We believe that, by making clear the translation of arbitrary CASP programs to SMT, our work will boost the cross-fertilization between the two areas.