Smoothed complexity for Pseudoboolean solver heuristics on industrial TSP instances.

Tamkin Avi Khan, Antonina Kolokolova

Memorial University of Newfoundland

Abstract. SAT solvers have been surprisingly successful in solving instances of NP-hard problems coming from industrial settings such as scheduling and verification. In this paper, we investigate their usefulness for solving instances of the asymmetric TSP problem resulting from routing of autonomous underwater vehicles in the ocean, with the goal of understanding which properties of this class of instances make the problem easier or harder for the solvers.
In particular, we analyse performance of PseudoBoolean Optimization solvers on a class of asymmetric TSP instances coming from the problem of routing gliders (small autonomous underwater vehicles that are especially susceptible to the influence of currents). We use the standard linear programming encoding of the TSP instances [MTZ60] as a basis for our CNF and pseudoboolean encodings. We then compare the instances obtained from this problem to asymmetric TSP on a random directed graph, as well as Euclidean and Euclidean with various amounts of noise. The Euclidean with noise setting suggests the smoothed analysis of Spielman and Tang [ST04] as a natural framework to compare the performance of the solvers. Here, we look at heuristics that the PBO solvers use and relate them in terms of the smoothed complexity of TSP with respect to them.

1 Introduction

Over the last several years, generic constraint satisfaction problem solvers, in particular SAT solvers, became a staple method for solving problems arising in practice, from automated hardware and software verification to AI planning problems, to exam scheduling at universities. They seem to perform surprisingly well on the instances of NP-hard problems coming from a practical, real-world setting, yet the reasons for this spectacular performance are not well understood. The heuristics that the solvers use seem to "pick up" on some regularities in the instances that are not explicit enough for us to design efficient algorithms for these classes of instances directly.

Investigating which properties of these industrial, real-world instances make the problems easier for the solvers has been an active area of research for many years. There is a number of publications identifying such properties of real-world instances. For example, [ABL09] shows that the underlying graphs of many industrial instances exhibit scale-free behaviour (there is a power law distribution of the variable occurrence frequencies), which may account for the success of SAT solvers on this class of instances. A different characteristic of the "easiness" of an instance is an existence of a small backdoor [WGS03], that is, a set of variables such that setting them makes the resticted formula easy (e.g., a Horn formula); in some planning instances the backdoors were found to be very small. Yet another "island of tractability" corresponds to formula in which underlying graphs are acyclic (or, more generally, have bounded treewidth or pathwidth); starting from [Fre85,DP89], there have been numerous extensions and combinations of these characteristics with other parameters.

Here, we will focus our attention on the class of problems which so far has been hard for SAT solvers. Moreover, we add a level of complexity by encoding not just a combinatorial problem (which, in this case, would be the Hamiltonian Cycle), but rather a numerical version of it, TSP. This naturally leads us to the realm of Pseudo Boolean Constraint solvers, and Satisfiability Modulo Theory. Our goal is to see whether, independent of the encoding and heuristics used by the solvers, there are some properties of the underlying instances which might make the problem easier or harder for the solvers (PBO or SMT solvers, in this case).

One such parameter is symmetry: is it easier for solvers to work with an instance which inherently is more symmetric, or does breaking a symmetry (for example, by adding noise) makes the problem more amenable to the heuristics, possibly by eliminating local minima? The experimental results seem to indicate that a more symmetric instance (in our case, TSP on a Euclidean graph) presents more of a challenge to the PBO solvers we tried; however adding even a small amount of noise to the weights allow the solvers to run faster and significantly bring down the number of decisions. Another question is whether this behaviour is preserved between different solvers (i.e., heuristics) and encodings of the problem.

The choice of the problem and instances came from an industrial project on mission planner for ocean gliders. The gliders (in particular, SLOCUM gliders) are small autonomous underwater vehicles. The name "gliders" comes from their way of locomotion: they have a chamber which they can fill with water or empty out. When the chamber is empty, the glider rises to the surface; with the chamber full, it sinks to the bottom of the ocean. As it ascends, positioning the glider at an angle allows it to move forward in a chosen direction. They are not very powerful and very much affected by the currents, so planning a mission requires knowledge of the weather, ocean currents, tides and so on. But in its simplest form, the mission plan can be viewed as an asymmetric Traveling Salesperson Problem instance. More precisely, given a sequence of goal points in the ocean and a starting point, a mission plan will consist of an ordering of the goal points so that the glider can visit all of them within a time limit (determined by its battery life), if it is possible. In general, the underlying graph weights are time-varying (in particular, tidal currents need to be taken into account: at a wrong time of the day, a glider might be physically incapable of moving away to the shore or coming back). Also, the number of gliders can be variable, and there can be a number of other, combinatorial constraints imposed on the problem. This lead us to investigate whether SAT/SMT/PBO solvers can be a viable tool for such mission planning task (provided that the underlying graph is computed using other techniques). Additionally, it gave us an opportunity to experiment with performance of the solvers on the data that comes from the missions planned on the actual maps of the currents, with the given parameters of the gliders.

1.1 Specialized heuristics for solving TSP

The optimization version of the Traveling Salesperson Problem (also known as the Traveling Salesman Problem) is formulated as a problem of finding, in a complete graph with edge weights, a tour (that is, a simple cycle containing all vertices) of the minimal weight, where weight of a tour is the sum of weights of its edges. The decision version asks for an existence of a tour of weight smaller than a given bound; there is a simple reduction from Hamiltonian Circuit problem to the decision TSP, giving its NP-hardness (just set the bound to be 0, and set all and only edges of a graph to be of 0 weight, with the rest of weights being any small positive number, e.g. 1).

Over the years, a significant body of work was accumulated on various heuristics for solving TSP, and its easier variants. In partiular, approximation algorithms for TSP are a rich area, with numerous heuristics for variants of TSP. Notably, TSP on Euclidean graphs (that is, graphs in which vertices are points on a plane, and the weights are a Euclidean distance between them) can be approximated with arbitrary precision, and metric TSP (where weights satisfy triangular inequality) can be approximated to within 1.5 the optimal by using Christofides' algorithm [Chr76]. However, for the linear programming formulation of TSP, there are constant integrality gaps results for the linear programming relaxation approaches, even for TSP with triangle inequality [Wat11].

TO BE CONTINUED...

1.2 Smoothed analysis

1.3 Our Results

2 Preliminaries

2.1 TSP

2.2 SAT and Pseudo Boolean Constraints solvers

3 Experimental results

4 Analysis

5 Conclusions

References

[ABL09] Carlos Ansótegui, Maria Luisa Bonet, and Jordi Levy. On the structure of industrial sat instances. In *CP*, pages 127–141, 2009.

[Chr76] Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, DTIC Document, 1976.

[DP89] Rina Dechter and Judea Pearl. Tree clustering for constraint networks. *Artificial Intelligence*, 38(3):353–366, 1989.

[Fre85] Eugene C Freuder. A sufficient condition for backtrack-bounded search. *Journal of the ACM (JACM)*, 32(4):755–761, 1985.

[MTZ60] C. E. Miller, A. W. Tucker, and R. A. Zemlin. Integer programming formulations of traveling salesman problems. *Journal of the Association for Computing Machinery*, 7:326–329, 1960.

[ST04] Daniel A Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM (JACM)*, 51(3):385–463, 2004.

[Wat11] Thomas Watson. Lift-and-project integrality gaps for the traveling salesperson problem. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 18, page 97. Citeseer, 2011.

[WGS03] Ryan Williams, Carla P Gomes, and Bart Selman. Backdoors to typical case complexity. *INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE*, 18:1173–1178, 2003.