Adv. Studies Theor. Phys., Vol. 6, 2012, no. 25, 1235 - 1238

SAT Solvers for the Problem of Sensor Placement

Anna Gorbenko

Department of Intelligent Systems and Robotics
Ural Federal University
620083 Ekaterinburg, Russia
gorbenko.ann@gmail.com

Vladimir Popov

Department of Intelligent Systems and Robotics
Ural Federal University
620083 Ekaterinburg, Russia
Vladimir.Popov@usu.ru

Abstract

In this paper we consider an approach to solve the problem of sensor placement. This approach is based on constructing SAT solvers for logical models of the problem.

PACS: 42.30.Tz

Keywords: sensor placement, logical models, SAT solvers, 3SAT

Many different formalizations of the problem of sensor placement received a lot of attention recently (see e.g. [1, 2]). Note that sensor placement is extensively used for improved robotic navigation (see e.g. [3, 4, 5]). In particular, visual landmarks problems are extensively studied in contemporary robotics (see e.g. [6, 7, 8, 9]). In this paper we consider SAT solvers for logical models of SP problem (see [2]).

In papers [10, 11, 12, 13, 14] the authors considered some algorithms to solve logical models (see also [15, 16, 17]). Also, we have obtained explicit reductions from SP to MAXSAT, SAT and 3SAT (see [2, 18]).

We use algorithms fgrasp and posit from [19]. Also we design our own genetic algorithm for SAT which based on algorithms from [19].

Consider a boolean function $g(x_1, x_2, \ldots, x_n) = \wedge_{i=1}^{m} \mathcal{C}_i$, where $m \geq 1$, and each of the $\mathcal{C}_i$ is the disjunction of one or more literals. Let $|\mathcal{C}_i|$ be a number of literals in $\mathcal{C}_i$. Let $occ(x_i, g)$ be a number of occurrences of x_i in g. Respectively, let $occ(\neg x_i, g)$ be a number of occurrences of x_i in g. For example, if $g = (x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_4) \wedge (\neg x_1 \vee x_5)$, then $occ(x_1, g) = 2$, $occ(\neg x_1, g) = 1$.

We consider a number of natural principles that define importance of a variable x_i for satisfiability of boolean function g. These principles suggest us correct values of variables.

1. If $occ(x_i, g) \geq 0$ and $occ(\neg x_i, g) = 0$, then $x_i = 1$.

2. If $occ(x_i, g) = 0$ and $occ(\neg x_i, g) \geq 0$, then $x_i = 0$.

3. If $x_i = \mathcal{C}_j$ for some j, then $x_i = 1$.

4. Given positive integers $p_1, p_2, \ldots, p_m, q_1, q_2, \ldots, q_m$ and a set of rational numbers $\{\alpha_{i,u}, \beta_{i,v} \mid 1 \leq i \leq m, 1 \leq u \leq p_i, 1 \leq v \leq q_i\}$. If

$$\sum_{1 \leq j \leq m, 1 \leq u \leq p_j, |\mathcal{C}_j| = u} \alpha_{j,u} occ(x_i, \mathcal{C}_j) \geq \sum_{1 \leq j \leq m, 1 \leq v \leq q_j, |\mathcal{C}_j| = v} \beta_{j,v} occ(\neg x_i, \mathcal{C}_j),$$

then $x_i = 1$.

Based on these principles, we can consider the following four types of commands: P_1, P_2, P_3, P_4. Also we consider the following three commands for run algorithms: Try_fgrasp, Try_posit, and Try_ga, where Try_ga runs a simple genetic algorithm.

Denote by $\mathcal{R}$ the set of commands of these seven types. Arbitrary element of $\mathcal{R}^*$ it is possible to consider as a program for finding values of variables of a boolean function. We assume that such programs are executed on a cluster.

Execution of each of commands of type P_i reduces the number of variables of a boolean function by one. Execution of each of commands Try_fgrasp, Try_posit, and Try_ga consists in the run of corresponding algorithm for current boolean function on a separate set of calculation nodes and the transition to the next command.

Algorithms fgrasp and posit we run only on one calculation node. Genetic algorithms can be used in parallel execution. We use auxiliary genetic algorithm which determine the number of calculation nodes.

Initially, we selected a random subset of $\mathcal{R}^*$. We use a genetic algorithm to select a program from the current subset of $\mathcal{R}^*$ and a genetic algorithm for evolving the current subset of $\mathcal{R}^*$. The evolution of the current subset of $\mathcal{R}^*$ implemented on a separate set of calculation nodes. For every subsequent boolean functions it is used the current subset of $\mathcal{R}^*$ which is obtained by taking into account the results of previous runs.

We use heterogeneous cluster based on three clusters (Cluster USU, Linux, 8 calculation nodes; umt, Linux, 256 calculation nodes; um64, Linux, 124 calculation nodes) [20].

Algorithms fgrasp and posit used only for 3SAT. For SAT and MAXSAT used simple genetic algorithm (SGA), and our algorithm (OA). We create a generator of natural instances for SP. We consider instances with N and S from 400 to 600. Selected experimental results are given in Figures 1 – **??**.

time	fgrasp	posit	SGA	OA
average	56 min	1.03 h	1.26 h	27.38 min
max	18.43 h	16.67 h	28.25 h	13.59 h
best	3.52 min	5.12 min	3.32 min	28 sec

Figure 1: Experimental results for 3SAT

time	SGA SAT	OA SAT	SGA MAXSAT	OA MAXSAT
average	1.17 h	53.12 min	1.14 h	49.29 min
max	26.29 h	17.33 h	32.84 h	25.17 h
best	3.18 min	16 sec	1.09 min	12 sec

Figure 2: Experimental results for SAT and MAXSAT

References

[1] A. Gorbenko, M. Mornev, V. Popov, and A. Sheka, The problem of sensor placement for triangulation-based localisation, *International Journal of Automation and Control*, 5 (2011), 245-253.

[2] A. Gorbenko, M. Mornev, V. Popov, and A. Sheka, The Problem of Sensor Placement, *Advanced Studies in Theoretical Physics*, 6 (2012), 965-967.

[3] A. Gorbenko, A. Lutov, M. Mornev, and V. Popov, Algebras of Stepping Motor Programs, *Applied Mathematical Sciences*, 5 (2011), 1679-1692.

[4] A. Gorbenko, V. Popov, and A. Sheka, Robot Self-Awareness: Temporal Relation Based Data Mining, *Engineering Letters*, 19 (2011), 169-178.

[5] A. Gorbenko and V. Popov, Anticipation in Simple Robot Navigation and Learning of Effects of Robot's Actions and Changes of the Environment, *International Journal of Mathematical Analysis*, 6 (2012), 2747-2751.

[6] A. Gorbenko and V. Popov, On the Problem of Placement of Visual Landmarks, *Applied Mathematical Sciences*, 6 (2012), 689-696.

[7] A. Gorbenko and V. Popov, A Real-World Experiments Setup for Investigations of the Problem of Visual Landmarks Selection for Mobile Robots, *Applied Mathematical Sciences*, 6 (2012), 4767-4771.

[8] A. Gorbenko and V. Popov, The Problem of Selection of a Minimal Set of Visual Landmarks, *Applied Mathematical Sciences*, 6 (2012), 4729-4732.

[9] A. Gorbenko and V. Popov, Computational Experiments for the Problem of Selection of a Minimal Set of Visual Landmarks, *Applied Mathematical Sciences*, 6 (2012), 5775-5780.

[10] A. Gorbenko, M. Mornev, and V. Popov, Planning a Typical Working Day for Indoor Service Robots, *IAENG International Journal of Computer Science*, 38 (2011), 176-182.

[11] A. Gorbenko and V. Popov, Programming for Modular Reconfigurable Robots, *Programming and Computer Software*, 38 (2012), 13-23.

[12] A. Gorbenko and V. Popov, The set of parameterized k-covers problem, *Theoretical Computer Science*, 423 (2012), 19-24.

[13] A. Gorbenko, V. Popov, and A. Sheka, Localization on Discrete Grid Graphs, *Lecture Notes in Electrical Engineering*, 107 (2012), 971-978.

[14] A. Gorbenko and V. Popov, Task-resource Scheduling Problem, *International Journal of Automation and Computing*, 9 (2012), 429-441.

[15] A. Gorbenko and V. Popov, The Longest Common Parameterized Subsequence Problem, *Applied Mathematical Sciences*, 6 (2012), 2851-2855.

[16] A. Gorbenko and V. Popov, The Binary Paint Shop Problem, *Applied Mathematical Sciences*, 6 (2012), 4733-4735.

[17] A. Gorbenko and V. Popov, On the Longest Common Subsequence Problem, *Applied Mathematical Sciences*, 6 (2012), 5781-5787.

[18] A. Gorbenko and V. Popov, On the Problem of Sensor Placement, *Advanced Studies in Theoretical Physics*, 6 (2012), 1117-1120.

[19] URL: http://people.cs.ubc.ca/~hoos/SATLIB/

[20] URL: http://parallel.uran.ru/mvc_now/hardware/supercomp.htm

Received: September, 2012