Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Note

Encoding cardinality constraints using standard encoding of generalized selection networks preserves arc-consistency

Michał Karpiński

Institute of Computer Science, University of Wrocław, Joliot-Curie 15, 50-383 Wrocław, Poland

A R T I C L E I N F O

Article history:
Received 31 January 2017
Received in revised form 26 September 2017
Accepted 29 September 2017
Available online xxxx
Communicated by R. Klasing

Keywords:
Cardinality constraints
Comparator networks
Selection networks
Arc-consistency
SAT encodings

A B S T R A C T

Cardinality constraints state that at most (at least, or exactly) k out of n propositional variables can be true. In this paper we prove the arc-consistency property of an encoding of cardinality constraint (into a CNF formula) that we call a standard encoding of generalized selection networks.

1. Introduction

Cardinality constraints are of the form $x_1 + \cdots + x_n \sim k$, where $x_1, \ldots, x_n$ are Boolean variables (or their negations), k is a natural number, and $\sim$ is a relation from the set $\{=, \leq, <, \geq, >\}$. Such constraints appear naturally in formulations of different real-world problems like timetabling [3], formal hardware verification [5] or cumulative scheduling [13]. Those problems (and many others) are hard to solve but their instances can be reduced to a series of SAT instances with additional sets of cardinality constraints. One of possible ways to solve such instances is to use a SAT-solver, therefore efficient encodings of cardinality constraints into CNF formulas are a crucial part of their applicability.

Comparator networks are simple data-oblivious models for sorting-related algorithms. In recent years several new encodings of cardinality constraints were proposed that are based on comparator networks. The classic *odd–even* sorting networks by Batcher [4] were used in [1,2,7] whereas *pairwise* sorting networks by Parberry [12] were used in [6,9]. It has been observed that using selection networks instead of sorting networks is more efficient for the encoding of cardinality constraints. The output of a selection network is the k largest elements from n inputs. Additionally the output has to be sorted. We can enforce the constraint $x_1 + \cdots + x_n < k$ by first building a selection network with input variables $\{x_1, \ldots, x_n\}$ and output variables $\{y_1, \ldots, y_k\}$, then setting the output variable y_k to 0. With this, no more than $k - 1$ x_i's can be set to 1. In this paper we consider only the "$<$" relation. Constraints using other relations can be reduced to the one above (see [2]).

There is a property of an encoding called (generalized) arc-consistency which – in the case of cardinality constraints and SAT-solvers – states that: for a constraint $x_1 + \cdots + x_n < k$, as soon as $k - 1$ variables among the x_i's become true, unit propagation sets all other x_i's to false. This has a positive impact on the practical efficiency of SAT-solvers, which is an important factor for the Constraint Programming community. The encodings proposed in [1,2,6,7,9] are all arc-consistent.

E-mail address: karp@cs.uni.wroc.pl.

https://doi.org/10.1016/j.tcs.2017.09.036

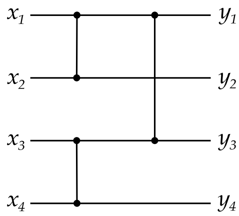

Fig. 1. Comparator network.

Motivation: In [1,2,6] the authors use properties of their constructions to prove arc-consistency which is usually long and technical. In [9] the authors relieve some of this burden for the future researchers by proving that the standard encoding of any selection network preserves arc-consistency. In this paper we would like to generalize their proof to the extended model of selection networks. The motivation behind this comes from [1] where authors observe that having smaller networks in terms of number of comparators is not always beneficial in practice, as it should also be accompanied with a reduction of SAT-solver run-time. They extend the classic comparator network model by mixing the **Direct Cardinality Networks** (DCNs) into their recursive construction, for small values of n and k. The main building blocks of DCNs are m-Cardinality Networks which select m largest (sorted) values from the input of size n directly (without using auxiliary variables) and can be viewed as a certain generalization of a simple comparator. We will call them m-selectors for short. This approach reduces the number of variables in exchange for increased number of clauses. Experiments show that this approach is very competitive. Knowing this, we anticipate that more constructions will emerge that will generalize the comparator networks further and instead of using simple comparators (2-sorters) they will use sorters of higher order as building blocks (potentially mixed with m-selectors). We assume without loss of generality, that those networks consists of only selectors. We can do this because sorter is a special case of selector. This will leave us with fewer number of cases in the proofs. Such networks will be called **Generalized Selection Networks** (GSNs). Some encodings based on this new approach are already available, for example, see [10].

Structure of the paper: The rest of the paper is organized as follows: Section 2 contains definitions and notations used in the paper. In Section 3 we prove arc-consistency of standard encoding of GSN and we give concluding remarks in Section 4.

2. Preliminaries

We begin by briefly introducing comparator networks. Traditionally comparator networks are presented as circuits that receive n inputs and permute them using comparators (2-sorters) connected by "wires". Each comparator has two inputs and two outputs. The "upper" output is the maximum of inputs, and "lower" one is the minimum. The standard definitions and properties of comparator networks can be found, for example, in [11]. The only difference is that we assume that the output of any sorting operation or comparator is in a non-increasing order.

Example 1. Fig. 1 is an example of a simple comparator network consisting of 3 comparators. It outputs the maximum from 4 inputs on the top horizontal line, namely, $y_1 = \max\{x_1, x_2, x_3, x_4\}$.

We are interested in using comparator networks in the context of Boolean formulas, therefore we limit the domain of the inputs to 0-1 values.

Definition 1 *(sequences).* A binary sequence of length n is a sequence of 0-1 numbers $\bar{x} = \langle x_1, \dots, x_n \rangle$, where $x_i \in \{0, 1\}$, $1 \le i \le n$. We say that a binary sequence $\bar{x} \in \{0, 1\}^n$ (of length $|\bar{x}| = n$) is sorted if $x_i \ge x_{i+1}$, $1 \le i < n$. The number of ones in $\bar{x}$ is denoted by $|\bar{x}|_1$.

A clause is a disjunction of literals (Boolean variables x or their negation $\neg x$). A CNF formula is a conjunction of clauses. We introduce the convention, that $\langle x_1, \dots, x_n \rangle$ will denote the input and $\langle y_1, \dots, y_n \rangle$ will denote the output of some order n comparator network (or GSN). We would also like to view them as sequences of Boolean variables, that can be set to either true (1), false (0) or undefined (X).

Unit Propagation (UP) is a process, that for given CNF formula, clauses are sought in which all literals but one are false (say l) and l is undefined (initially only clauses of size one satisfy this condition). This literal l is set to true and the process is iterated until reaching a fix point.

2.1. The Generalized Selection Network

Here we formally define the main part of our encoding, which is the Generalized Selection Network (GSN). The reader is encouraged to check other papers that use high-order sorters as components in the construction of sorting networks (for example, [14] and [8]) to gain better understanding of GSN.

Please cite this article in press as: M. Karpiński, Encoding cardinality constraints using standard encoding of generalized selection networks preserves arc-consistency, Theoret. Comput. Sci. (2017), https://doi.org/10.1016/j.tcs.2017.09.036