Declarative Specification and Solution of Combinatorial Auctions Using Logic Programming

Chitta Baral and Cenk Uyan

Department of Computer Sc. and Engg., Arizona State University
Tempe, Arizona 85287
{chitta,cuyan}@asu.edu

Abstract. In a combinatorial auction problem bidders are allowed to bid on a bundle of items. The auctioneer has to select a subset of the bids so as to maximize the price it gets, and of course making sure that it does not accept multiple bids that have the same item as each item can be sold only once. In this paper we show how the combinatorial auction problem and many of its extensions can be expressed in logic programming based systems such as Smoels and dlv. We propose this as an alternative to the standard syntax specific specialized implementations that are much harder to modify and extend when faced with generalizations and additional constraints.

1 Introduction and Motivation

In a simple auction several bidders bid for an item and the auctioneer selects the highest bid. Often bidders need a bundle of items, where the worth of the whole bundle – to the bidder – may be more than the sum of the individual worth of each item in the bundle. For example, let A and B be two adjacent real estate plots. A single developer can often make more money developing both plots together than two different developers developing A and B separately without co-operating with each other. This happens if say both A and B are needed to create a lucrative golf course while A and B separately can only be used for less profitable purposes. The opposite may be true in some cases too. The cases that are often mentioned with regards to both are airport landing slots [9], bandwidth auctions, real estate auctions, and transportation exchanges [11].

In such cases participating in parallel or sequential auctions for each item in a bundle desired by a bidder is risky as the bidder may not win all items in the bundle. Moreover it would be difficult for him to individually price each item in the bundle. One way to avoid such problems is to have combinatorial auctions where bidders are allowed to bid on bundles. Although this is good for the bidder, the seller’s problem of deciding which bids to accept becomes harder, as different bidders can make up their own bundle on which they bid on.

Recently, there has been a lot of interest in this problem because of its applicability in Internet based auctions, B2B exchanges, and multi-agent systems [16,7]. There have been several papers [12,13,2,4,6,15,10] that analyze this
problem and present algorithms and techniques to solve it and a few of its generalizations. One starting point that guides research on this is the result from [10] which shows the problem of finding the optimal set of bids (that maximize the seller’s take) to be NP-Complete.

So far there are three different approaches for solving this problem: complete algorithms [12,2,10] that find an optimal solution in the general case, incomplete methods [4] that find high quality solutions quickly, and identification of tractable special cases and algorithms for those cases [15,13]. The other possible approach of finding approximation algorithms is blocked by the result from [12] that shows that no polynomial algorithm can guarantee a solution that is close to optimal.

In this paper we follow the first approach of obtaining optimal solutions in the general case. Our methodology is different from the earlier approaches [12,2,10] in that we would like to represent the problem in a declarative knowledge representation language such that optimal ‘models’ of the representation correspond to optimal solutions. This is similar to the methodology of satisfiability based planning [5] where the planning problem is represented as a propositional theory, and each model of this theory encodes a plan. The main motivation behind our approach of using a declarative knowledge representation language is that we would like the process of adding additional constraints, or making a generalization to be easier. This differs from the other approaches [12,13,2,6] where major changes were needed to move from single unit combinatorial auctions to multi-unit combinatorial auctions. Also, as mentioned in [13] additional generalizations necessitates change in the code, which requires the knowledge of the structure of the code and hence can only be done by people adequately familiar with the original code. In contrast we will show that when using a declarative knowledge representation language additional generalization, or addition of new constraints often leads to adding a few extra rules, without needing the detailed knowledge of the original code or its structure.

The declarative language that we will use throughout this paper is Smodels [8,14], an extension of logic programming with answer set semantics [3]. It has new constructs such as cardinality and weight constraints, and optimization statements. It is preferable over propositional logic as it is more expressive in terms of being able to express transitive closure, causality, and aggregation. Moreover, it is a non-monotonic language and hence more suitable for knowledge representation and finally it includes optimization statements. (A more detailed argument about the advantages of logic programming with answer set semantics over other logics is given in the draft of a book by the first author available at

1 Strictly speaking, Smodels is a system that started of as implementing the answer set semantics of logic programs and now has several new constructs. By the Smodels language we refer to the extension of logic programs that is used by the Smodels system.

We would like to mention that some of the encodings in this paper can also be expressed in the language of the dlv system [1]. Due to lack of space we only focus on the Smodels system.