The Temporal Knapsack Problem and Its Solution

Mark Bartlett[1], Alan M. Frisch[1], Youssef Hamadi[2], Ian Miguel[3], S. Armagan Tarim[4], and Chris Unsworth[5]

[1] Artificial Intelligence Group, Dept. of Computer Science, Univ. of York, York, UK
[2] Microsoft Research Ltd., 7 J J Thomson Avenue, Cambridge, UK
[3] School of Computer Science, University of St Andrews, St Andrews, UK
[4] Cork Constraint Computation Centre, Univ. of Cork, Cork, Ireland
[5] Department of Computing Science, University of Glasgow, UK

Abstract. This paper introduces a problem called the temporal knapsack problem, presents several algorithms for solving it, and compares their performance. The temporal knapsack problem is a generalisation of the knapsack problem and specialisation of the multidimensional (or multiconstraint) knapsack problem. It arises naturally in applications such as allocating communication bandwidth or CPUs in a multiprocessor to bids for the resources. The algorithms considered use and combine techniques from constraint programming, artificial intelligence and operations research.

1 Introduction

This paper defines the temporal knapsack problem (TKP), presents some algorithms for solving it and compares the performance of the algorithms on some hard instances. TKP is a natural generalisation of the knapsack problem and a natural specialisation of the multi-dimensional knapsack problem. Nonetheless, it is—as far as we know—a new problem.

In the TKP a resource allocator is given bids for portions of a timeshared resource — such as CPU time or communication bandwidth — or a shared-space resource — such as computer memory, disk space, or equivalent rooms in a hotel that handles block-booking. Each bid specifies the amount of resource needed, the time interval throughout which it is needed, and a price offered for the resource. The resource allocator will, in general, have more demand than capacity, so it has the problem of selecting a subset of the bids that maximises the total price obtained.

We were initially drawn to formulating the TKP from our interest in applying combinatorial optimisation techniques in the context of grid computing.

R. Barták and M. Milano (Eds.): CPAIOR 2005, LNCS 3524, pp. 34–48, 2005.

Applications that use a grid simultaneously require different resources to perform large-scale computations. An advanced-reservation system will be used to guarantee a timed access to resources through some service level agreement [1].

The agreement is reached via negotiation, where end users present reservation bids to resource providers. Each bid specifies the resource category, start time, end time and required quality of service (e.g., bandwidth, number of nodes) [2]. If end users offer a price they are willing to pay for the resource, advanced reservation allocation in a grid infrastructure becomes equivalent to the TKP.

Our algorithms are designed to be used by a resource provider to select efficiently the right subset of customers with respect to resource requirements and the provider's utility. So far advanced reservation is not used in grid infrastructures, which still use specialised *fifo* scheduling policies inherited from high-performance computing. However, the convergence between web services and grid computing, combined with the arrival of the commercial grid, make the efficient use of valuable resources critical [3]. Our algorithms fit well into next-generation grids and represent the first attempts towards efficient grid resource schedulers.

2 The Temporal Knapsack Problem

A formal statement of the TKP is given in Figure 1. Here, and throughout, $bids(t)$ is $\{b \in bids | t \in duration(b)\}$. It is important to notice that TKP is *not* a scheduling problem.

Figure 2(a) illustrates an instance of TKP that has seven bids, $b_1, \ldots, b_7$, and 10 times, $t_1, \ldots, t_{10}$, which are displayed on the x-axis. The instance has a uniform capacity of 10, which is not shown. The optimal solution to this instance is to accept bids b_1, b_4, b_5, and b_6, yielding a total price of 22.

The traditional knapsack problem, as overviewed by Martello and Toth [4], is a special case of TKP in which there is only a single time. Since the knapsack problem, which is NP-hard, is a special case of TKP, TKP is also NP-hard.

Given:	$times$, a finite, non-empty set totally ordered by $\leq$
	for each $t \in times$, $capacity(t)$, a positive integer
	$bids$, a finite set
	for each $b \in bids$,
	$price(b)$, a positive integer
	$demand(b)$, a positive integer
	$duration(b) = [start(b), end(b)]$, a non-empty interval of $times$
Find:	a set $accept \subseteq bids$
Such that:	$\forall t \in times$, $\sum_{b \in (accept \cap bids(t))} demand(b) \leq capacity(t)$
Maximising:	$\sum_{b \in accept} price(b)$

Fig. 1. Definition of the temporal knapsack problem