Experiences With Scheduling and Mapping Games for Adaptive Distributed Systems: Summary

Bin Lin Peter A Dinda
Department of Electrical Engineering and Computer Science
Northwestern University
{b-lin, pdinda}@northwestern.edu

ABSTRACT

We apply the concept of "games with a purpose" to NP-complete mapping and scheduling problems in distributed systems and report our experiences. The particular context is a scheduling and mapping problem that emerges when running parallel applications in a distributed virtualized computing environment, specifically BSP applications in our Virtuoso system. We describe the design and implementation of an interface that presents the problem as a game, and a user study we undertook to evaluate the interface. The results show that there is promise—at least at small scales, most of our naive users are able to find solutions that are reasonable.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]; C.2.4 [**Distributed Systems**]; H.5.2 [**User Interfaces**]

General Terms

Virtualization, Parallelism, Human Factors

Keywords

Optimization, User Studies, Games With A Purpose

1. INTRODUCTION

Optimization problems emerge from distributed and parallel computing systems. These problems are often NP-hard or NP-complete, or at least difficult to solve efficiently. In some situations, such as in interactive computing, they are even difficult to *pose* well. In this paper, we focus on the solution of fully posed, but difficult to *solve* optimization problems. We consider whether such problems can be solved by naive users if properly presented.

We propose a technique called *human-driven search* to address such problems. Human-driven search uses direct human input, from naive users, to guide the search for a good solution, a valid configuration x that optimizes $f(x)$. We study human-driven search as an approach to *online* scheduling and mapping problems in distributed systems. Via a game interface, the user can continuously change schedules and mappings based on changes in performance or due to external load. The core questions we address are:

1 Is it possible to map a scheduling and mapping problem in distributed and parallel systems to a game that a naive user can play, with the side effect having of good game play correspond to a good solution to the problem?
2 Can naive users play such a game well?

We find that the answers to these questions are yes, at least for smale scale problems. A fully detailed treatment of this work, and work on a range of other examples of human-driven optimization, can be found elsewhere [1].

Effort sponsored by the National Science Foundation (NSF) via grants CNS-0720691, ANI-0301108, and EIA-0224449.

ICAC'09, June 15–19, 2009, Barcelona, Spain.
ACM 978-1-60558-564-2/09/06.

2. VIRTUOSO AND A PROBLEM

The Virtuoso project designs, implements, and evaluates systems software and middleware to support adaptive computing in virtualized distributed computing environments, including our own prototype. Virtuoso.cs.northwestern.edu and empathicsystems.org contain detailed information about our work. Virtuoso is intended to support a wide range of uses from interactive desktop users to batch parallel users. We focus on the latter in this paper.

In the present work, we have used trace-driven simulation. The Virtuoso simulator is a discrete event simulator that is driven by a trace collected from the execution of our Patterns bulk synchronous parallel (BSP) microbenchmark. The simulator faithfully captures the computation and communication within Virtuoso. The simulator provides an interface to an external agent (such as our game). The agent can interrupt the simulator (e.g., when the user seeks to interact with it).

Detailed formalizations of Virtuoso scheduling and mapping problems can be found on our earlier publications. Here we focus on a problem in which we seek to maximimze the performance of the Patterns BSP benchmark running in a collection of VMs. To do so, we can move VMs from host to host, and set their individual periodic real-time schedules.

3. A GAME FOR THE PROBLEM

We designed a game targeted at naive users, those who are unfamiliar with scheduling and mapping problems in distributed systems. It is necessary to present an *analogous* problem that is easily understood. Furthermore, any given game state the user reaches in solving this problem must be easily and unambiguously mappable back to a configuration of the system.

Figure 1 shows the interface that the game player sees. We present the concept of a VM's efficiency (percentage of

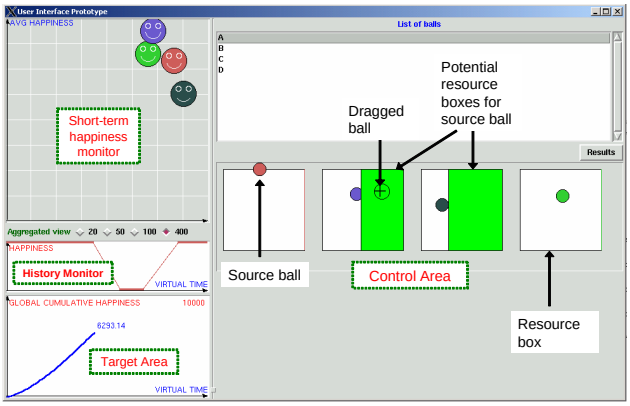

Figure 1: Interface for CPU scheduling & VM mapping game; the user is trying to migrate the ball in the left-most resource box to the second resource box to the left.

available compute time not spent waiting) as the happiness of a ball. $efficiency_{(k,i)}$ is shown as the "short term happiness" of ball k. The objective function $f(x)$ we seek to optimize is referred to as the "global cumulative happiness" of all of the balls, assuming all the balls are working together to make progress towards a global goal. In effect, the global cumulative happiness is the score.

The user plays for a fixed period of real time (and simulated time) with the goal of achieving the highest possible score. The user plays by dragging balls within boxes (VM schedule change), or between boxes (VM to host mapping (and schedule change, if desired)). As the user drags a ball, the game highlights and enforces the constraints on where the user may place it.

In the end of each game, the user will see a final screen that presents both the global cumulative happiness ($f(x)$), and its time average ($\frac{f(x)}{gameduration}$ as a percentage). When used in our user study, the screen also shows the highest possible score. The user is told that the closer his score to the highest possible score, the better he performed.

4. USER STUDY

We conducted a user study to evaluate how well naive users can play our game to solve the optimization problem. The 21 users in our study consisted of people with various backgrounds. Our test environment consisted of a desktop workstation which ran the game interface, and a server computer which ran the simulator.

For the tasks given in our evaluation, we can determine optimal solutions, either by construction or by simulation-based search. We also determine the average quality of a random solution through repeated simulation. We can thus compare the solutions derived from gameplay with optimal and random solutions. The study consists of two warm-up tasks and nine formal tasks.

4.1 Results

We discuss both qualitative and quantitative results for our study elsewhere [1, Chapter 7]. We summary high-level results below.

As far as can be judged with 21 users, there is considerable variation in user performance. Note that in almost all scales

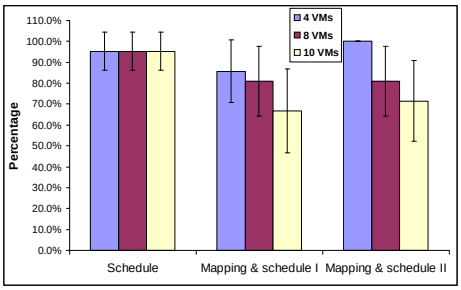

Figure 2: Percentage of users who find the optimal mapping; 95% confidence interval.

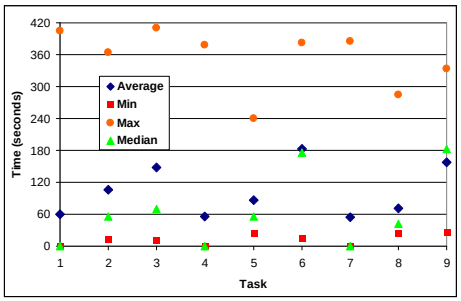

Figure 3: Duration to the optimal mapping.

and types of tasks considered, there are users who perform near-optimally. Interestingly, most users in our study are able to find optimal mappings. Figure 2 shows the percentage of users who successfully found optimal mappings and kept those mappings unchanged until the end of the task. In the worst case task we considered, more than 65% of users were able to find the optimal VM mapping.

The time to solution must be low in order for the game approach to scheduling and mapping problems to be workable. Figure 3 shows the statistics for the time to find optimal mappings. As task difficulty and problem scale grow, the average time to find the optimal mapping grows. However, users were able to find an optimal mapping in 2–3 minutes.

5. CONCLUSIONS

Our work investigated the two questions given in the introduction. We have found clear evidence that the answer to the first question is yes, while the second is a qualified yes. Our results show that most users are able to do well provided that the structure of the problem is simple enough, and its scale is relatively small. It is still unknown whether this is an inherent problem or a design issue. Also, user performance varies considerably. Even in the most difficult, and largest problems, some users perform well, independently of their background. This variation is to be expected for any game. An obvious question that emerges is how to identify and use these superior players.

6. REFERENCES

[1] Lin, B. *Human-driven Optimization*. PhD thesis, Northwestern University, Electrical Engineering and Computer Science Department, July 2007. NWU-EECS-07-04.