An Evolutionary Meta Hierarchical Scheduler for the Linux Operating System

Horst F. Wedde, Muddassar Farooq, and Mario Lischka

Informatik III, University of Dortmund,
44221, Dortmund, Germany
{wedde, farooq, lischka}@ls3.cs.uni-dortmund.de

Abstract. The need for supporting CSCW applications with *heterogeneous and varying user requirements* calls for adaptive and reconfigurable schedulers accommodating a mixture of real-time, proportional share, fixed priority and other policies, thus overcoming frustrating processor bottlenecks. In this paper we try to overcome this anomaly by proposing an *evolutionary strategy* for a *Meta Hierarchical Scheduler (MHS)* in which a *user is actively involved* in the design cycle of a scheduler. Our framework analyzes *user requirements* by formulating an *Abstract Scheduler Model (ASM)* for an optimum scheduler with the help of an evolutionary algorithm that satisfies the needs. Finally a C source code for this MHS is generated for the Linux kernel. Our experimental results demonstrate that our MHS enhances through the evolutionary approach, the user satisfaction level by a factor of two as compared to the satisfaction level achieved by the standard Linux scheduler.

1 The Evolutionary Algorithm in SAADI

The emergence of the Internet market toward the end of the last century has revolutionized the traditional use of PC. Nowadays a user in a CSCW environment attends a multimedia business meeting, take notes for minutes of the meetings and reads/sends important documents at the same time. In CSCW applications the processor usage pattern becomes frequently unpredictable because of changing user needs. Conflicting but coexisting scheduling requirements make it an uphill and difficult task to create a generic scheduler that could easily, efficiently if not optimally, adapt to the *user needs* in collaborating communities.
In SAADI framework we incorporate the *user needs* in the design cycle of a scheduler [2]. By using the constructs of SAADI-SDL, a scheduler description language, a user could specify the processing requirements of different applications and of their relative importance. Currently, the schedulable application set consists of single/multi-threaded batch, periodic (multimedia and network applications) and interactive applications. Our ASM module parses SAADI-SDL file and analyzes the requirements to generate scheduler individuals for the first generation of our evolutionary algorithm with the help of a context-free grammar. Figure 1 shows the complete design workflow in our SAADI framework. A scheduler individual consists of a hierarchical organization of simple schedulers as proposed by Regehr [1].

K. Deb et al. (Eds.): GECCO 2004, LNCS 3103, pp. 1334–1335, 2004.

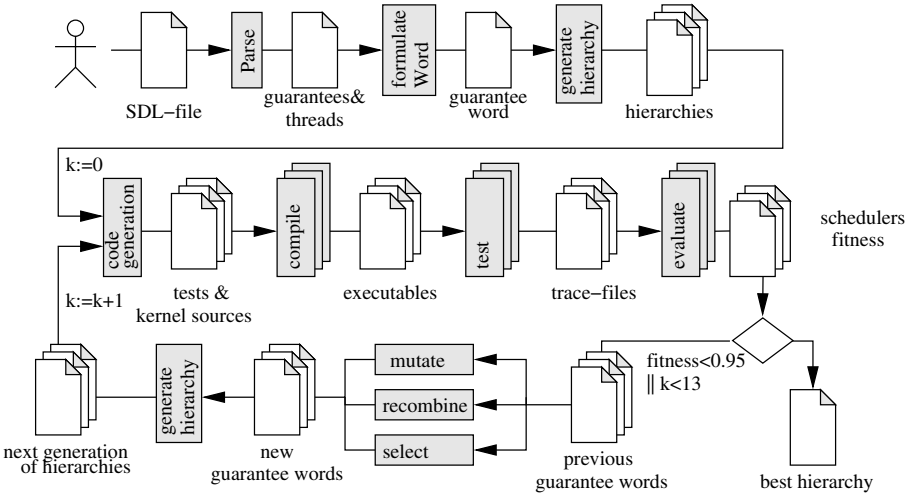

Fig. 1. Design Workflow in SAADI

2 Evaluation of Results and Future Work

We designed and developed a testing infrastructure to validate our approach. Here, we report the results for one SDL file in which the test user specified 7 interactive, 4 periodic and 4 batch threads. Our SAADI framework found an individual whose fitness value was 0.81. We also tested the standard Linux 2.5.58 scheduler under the same above-mentioned load . The fitness value of standard Linux scheduler was 0.33 on the average. This shows the superiority of SAADI over the standard Linux scheduler because the best performing individual of SAADI has 0.81 fitness value which is more than two times better than the standard Linux scheduler. Hence, this first step proved the conceptual validity of our model.

Currently SAADI provides an off-line interaction to a user. The user specifies his requirements using SDL, and then SAADI invokes a MHS generation process and the evolutionary algorithm to find an optimum MHS. In near future we intend to transform the framework into a real-time on-line module inside a kernel and provide easy system calls and a graphical user interface (GUI) for an administrative user to interact with the system.

References

1. John Regehr and John A. Stankovic. HLS: A framework for composing soft real-time schedulers. In *Proceedings of the 22nd IEEE Real-Time Systems Symposium (RTSS 2001)*, pages 3–14, London, UK, December 2001.
2. Horst F. Wedde, Muddassar Farooq, and Mario Lischka et. al. SAADI–An Integrated Approach of Adaptable Schedulers. Technical report-pg424, School of Computer Science, University of Dortmund, 2003.