Monitoring Program Behaviour on SUPRENUM

Publisher: IEEE Cite This PDF

M. Siegle ; R. Hofmann **All Authors**

Abstract:
It is often very difficult for programmers of parallel computers to understand how their parallel programs behave at execution time, because there is not enough insight into the interactions between concurrent activities in the parallel machine. Programmers do not only wish to obtain statistical information that can be supplied by profiling, for example. They need to have detailed knowledge about the functional behaviour of their programs. Considering performance aspects, they need timing information as well. Monitoring is a technique well suited to obtain information about both functional behaviour and timing. Global time information is essential for determining the chronological order of events on different nodes of a multiprocessor or of a distributed system, and for determining the duration of time intervals between events from different nodes. A major problem on multiprocessors is the absence of a global clock with high resolution. This problem can be overcome if a monitor system capable of supplying globally valid time stamps is used. In this paper, the behaviour and performance of a parallel program on the SUPRENUM multiprocessor is studied. The method used for gaining insight into the runtime behaviour of a parallel program is hybrid monitoring, a technique that combines advantages of both software monitoring and hardware monitoring. A novel interface makes it possible to measure program activities on SUPRENUM. The SUPRENUM system and the ZM4 hardware monitor are briefly described. The example program under study is a parallel ray tracer. We show that hybrid monitoring is an excellent method to provide programmers with valuable information for debugging and tuning of parallel programs.

Published in: [1992] Proceedings the 19th Annual International Symposium on Computer Architecture

Date of Conference: 19-21 May 1992

DOI: 10.1109/ISCA.1992.753329

Date Added to IEEE *Xplore*: 06 August 2002

Publisher: IEEE

Print ISBN:0-89791-509-7

Conference Location: Gold Coast, QLD, Australia

First Page of the Article

Monitoring Program Behaviour on SUPRENUM

Markus Siegle, Richard Hofmann,
Institut für Informatik VII, Universität Erlangen-Nürnberg,
Martensstraße 3, 8520 Erlangen, Germany,
e-mail siegle@immd7.informatik.uni-erlangen.de

Abstract

It is often very difficult for programmers of parallel computers to understand how their parallel programs behave at execution time, because there is not enough insight into the interactions between concurrent activities in the parallel machine. Programmers do not only wish to obtain statistical information that can be supplied by profiling, for example. They need to have detailed knowledge about the functional behaviour of their programs. Considering performance aspects, they need timing information as well. Monitoring is a technique well suited to obtain information about both functional behaviour and timing. Global time information is essential for determining the chronological order of events on different nodes of a multiprocessor or of a distributed system, and for determining the duration of time intervals between events from different nodes. A major problem on multiprocessors is the absence of a global clock with high resolution. This problem can be overcome if a monitor system capable of supplying globally valid time stamps is used.

In this paper, the behaviour and performance of a parallel program on the SUPRENUM multiprocessor is studied. The method used for gaining insight into the runtime behaviour of a parallel program is hybrid monitoring, a technique that combines advantages of both software monitoring and hardware monitoring. A novel interface makes it possible to measure program activities on SUPRENUM. The SUPRENUM system and the ZM4 hardware monitor are briefly described. The example program under study is a parallel ray tracer. We show that hybrid monitoring is an excellent method to provide programmers with valuable information for debugging and tuning of parallel programs.

Keywords: debugging, event-driven monitoring, multiprocessor, parallel program, performance evaluation, ray tracing, SUPRENUM, tuning.

1 Introduction

It is often very difficult for programmers of parallel computers to understand how their parallel programs behave at execution time, because there is not enough insight into the interactions between concurrent activities in the parallel machine. Programmers need to have detailed knowledge of the functional behaviour of their programs, and for the consideration of performance aspects they need timing information as well. Usually, methods such as profiling and accounting do not provide sufficient information, they only give summary statistical results. Therefore users often resort to rudimentary methods, such as writing log-files during program execution, in order to obtain debug information and performance information about their programs. But only a relatively small fraction of the needed information can be obtained that way. A major problem with multiprocessors is the absence of a global clock with high resolution. Global timing information is essential for determining the chronological order of events on different nodes of a multiprocessor or of a distributed system, and for determining the duration of time intervals between events from different nodes.

Facing this problem, our approach is to apply event-driven monitoring techniques [3] [8] [9] in order to find out how a parallel program behaves. In particular, we decided to use hybrid monitoring, which combines advantages of both hardware monitoring and software monitoring. Using software monitoring, it is relatively easy to relate the event traces obtained from the measurements to the measured program. But since monitoring is done within the object system (i.e. within the system under study), and therefore constitutes an extra workload, software monitoring changes the behaviour of the object system. Also, it is usually impossible to obtain global timing information because most parallel systems do not provide a global clock with high resolution. With hardware monitoring there is no intrusion and the timing problem can be solved by providing an external clock. But there is no easy way to relate the recorded signals to the source code of the measured program.

In hybrid monitoring, as in software monitoring, the program under study is instrumented by inserting additional instructions at points of interest. The execution of such a measurement instruction marks an event. It causes the output of measurement data, containing a token identifying the event and possibly some additional parameters, to an external interface. A hardware monitor is connected to the interface. It records the event stream coming from the interface, and stores the sequence of events together with the respective time stamps as an event trace. Since most of the work is done by the external hardware monitor, hybrid monitoring provides the capabilities of software monitoring at a much lower level of intrusion. The hardware monitor we use is a scalable distributed monitor system called ZM4. It is capable

