ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 2, February 2014

Survey on Compiler Analysis in India

Amruta Salunkhe, Prof.R.D.Bharati,

M.E. Student, Department of CSE, Dr.D.Y.Patil Institute Of Engg And Technology, University of Pune, Pune, India

Professor, Department of CSE, Dr.D.Y.Patil Institute Of Engg And Technology, University of Pune Pune, India

ABSTRACT: The key problem in achieving parallelism for distributed memory computers is data distribution, and achieving parallelism and load balancing. Data access latency, a limiting factor in the performance of distributed system, grows significantly with the number of cores in non-uniform cache architectures with distributed cache banks. Many different techniques and algorithms have been proposed or implemented for different programming environments, architectures and applications. However, there is little uniformity, common models, or approach to classifying and comparing such techniques and algorithms. This paper provides an overview of distributed system and how compiler based approach can be used to maximum data access locality and determine an optimize data placement.

KEYWORDS: Distributed System, Data Distribution, Compiler Analysis.

I. INTRODUCTION

Currently distributed system is widely used across the world by different companies to connect their various branches located at different geographical location. A distributed system is a software system in which collection of independent computers that appears to its users as a single computer system. With the use of distributed system, sharing of information and services, possibility to add components improves availability, performance, scalability, reliability and fault tolerance. Distributed system need radically different software than centralized system do.‖ data distribution in distributed system is a key factor in performance. Data access latency, a limiting factor in the performance of distributed system, grows significantly with the number of cores in non-uniform cache architectures with distributed cache bank. To mitigate this effect, we use a compiler based approach to leverage data access locality and determine an optimized data placement. The Data Distribution Service for Real-Time Systems is an Object Management Group machine to machine middleware standard that aims to enable high performance, scalable, dependable, real-time and interoperable data exchanges between publishers and subscribers. Data Distribution System is designed to address the needs of applications like financial trading, smart grid management, air traffic control and other big data applications. In applications such as smartphone operating systems, transportation systems and vehicles, software defined radio, and by healthcare providers, the standard is used. Data Distribution System plays a large role in the Internet of Things.

Multithreaded benchmarks shows that a large portion of data parallel applications tend to exhibit regularity in their data access patterns. Cache architectures for the last level on-chip cache for tiled CMPs typically fall into two classes, non-uniform cache access (NUCA) and private access with the higher-levels typically being private.

Detecting these patterns is critical to gain an insight of how data should be distributed among multiple CMP tiles to promote the locality of access within a tile and between tiles implied in the program. MMAPs are primarily related to array accesses and TI-Structures, there are other programming components affecting the actual access patterns.

The compilation methodology requires the use of several known compiler techniques that are well understood, such as symbolic analysis, constant propagation, expression folding, etc. We propose to use the compiler to determine the data partitioning implied by the program. Compiler analysis techniques to detect data access patterns, partitions, and communication patterns for multithreaded applications.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 2, February 2014

II. RELATED WORK

With the introduction of embedding a network in the cache on chip wire, delay problem will be solved in Non uniform cache access designs for future large integrated caches by embedding a network in the cache. Data migration within the cache is allowed in Non Uniform Cache Architecture by putting working set nearest the processor [1].

Princeton Application Repository[2] for Shared Memory Computers (PARSEC), a benchmark in studies of Chip Multiprocessors (CMPs). Benchmarks for multiprocessors earlier used to focus on high performance computing applications and synchronization methods were used in limited numbers. Upcoming applications in recognition, mining and synthesis (RMS) and systemsapplications are also considered in PARSE which simulates large scale multi-threaded commercial programs [2].

Analysis of energy models for on-chip interconnection networks and tradeoffs in tiled chip multiprocessors design. Using these models, investigation of the network architecture including topology, channel width, routing strategy, and buffer size done to analyze how it affects performance and impact area and energy efficiency. Advanced VLSI p-process is used for the implementation of tiled chip multiprocessors & Performance of on-chip networks designed for tiled chip multiprocessors is analyzed and compared to calculate area and energy efficiencies using this models[3].

Analysis also done on automatic parallelization system for Fortran programs, extensive experiments were carried out & the results were analyzed. The system implements an integrated collection of analyses consisting of dependence, privatization and reduction recognition for scalar variables & array variables, also scalar symbolic analysis is performed to support these [4].

III. METHODOLOGY

Distributed systems have following three significant characteristics:

- Concurrency of components
- Lack of a global clock
- Independent failure of components.

Goals of the distributed system can be better resources connectivity with users, transparency in data distribution, Openness and scalability.

A. Openness of Distributed Systems

Open distributed system should be able to interact with services from other open systems, irrespective of the environment where they reside. Portability of applications should be supported by systems & they should easily interoperate.

In order to achieve openness in distributed system, it should be made independent from following heterogeneity of the environment it resides.

- Hardware
- Platforms
- Languages

Examples

- The Internet: net of nets
- Intranets
- Mobile and ubiquitous computing.

B. Goals of Distributed Systems

Following are the goals of distributed systems

- Connecting resources and users
- Distribution transparency
- Openness
- Scalability

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 2, February 2014

C. Issues in Distributed System

There are various points that have to be taken into consideration while designing a distributed file system. The different issues are: transparency, flexibility, reliability, performance, scalability, Security which are described in this section.

1. Transparency

The most important design issue is to hide from the users the fact that processes and resources are physically distributed across the network. The different types of transparencies are as follows:

TABLE I

Transparency	Description
Access	Hides differences indata representation and invocation mechanisms
Location	Hides where an object resides
Migration	Hides ability of a system to change location of the object from an object
Relocation	Hides the ability of a system to change the location of an object from a client to which the client is bound
Replication	Hides the fact that an object or its state may be replicated and that replicas reside at different locations
Concurrency	Hides the coordination of activities between objects to achieve consistency at a higher level
Failure	Hides failure and possible recovery of objects
Persistence	Hides the fact that an object may be (partly) passivated by the system

2. Flexibility

The best way to achieve flexibility is to take a decision whether to use monolithic kernel or microkernel on each machine. The major functions of kernel are: memory management, process management and resource management. Monolithic kernels use the 'kernel does it all' approach with all functionalities provided by the kernel irrespective of whether all machines use it or not. On the other hand, micro-kernels use the minimalist, modular approach with accessibility to other services as needed.

3. Reliability

The users prefer a distributed system where multiple processes are available as it guards against single-processor system crashes. Thus on failure, a backup is available. Reliability means that data should be available without any errors. In case of replication, all copies should be consistent.

4. Performance

It means that an application should run just as it were running on a single processor. The metrics used for measuring performance are: response time, throughput, system utilization and amount of network capacity used. Message transmission over a LAN takes some time, about one millisecond. To optimize performance, reduce the number of messages transmitted. For example, a very small computation like addition of two numbers does no need to be computed remotely.

5. Scalability

Distributed systems are designed to work with a few hundred CPUs. There may be a need to expand the distributed system by adding more CPUs. To support more users or resources, there are limitations with centralized service, tables and algorithms. This scalability is related to its size.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 2, February 2014

6. Security

Security consists of three main aspects namely,

1). Confidentiality, which means protection against unauthorized access

2). Integrity, which implies protection of data against corruption

3). Availability, which means protection against failure and always being accessible.

D. Architectures in Distributed System

Analyzing the features that help to incorporate most appropriate and suitable distributed system.

Following different architecture exists in data distributed system

1). Client- Server Architecture: Sun Microsystems's Network System which provides standardized view of its local file system.

2). Symmetric Architecture: Based on peer-to-peer technology. In this file system, the clients also host the metadata manager code, resulting in all nodes understanding the disk structures.

3). Asymmetric Architecture: There are one or more dedicated metadata managers that maintain the file system and its associated disk structures. Examples include Luster and traditional NFS file systems.

4). Parallel Architecture: Across multiple storage devices on multiple storage servers, data blocks are banded in parallel. This supports concurrent read and write capabilities

E. Parallelism Detection

Parallelization using memory detection using memory classification analysis with memory access description.

The process of inter procedural parallelization the compiler build a phase description and iteration description for each array.

F. Data Distribution Selection

Data Distribution Selection step is organized in to two steps

1). The Locality Analysis Stage

This stage uses the information provided by the ID's to build the Locality Communication Graph(LCG) of the code. The LCG is our compile time representation of the locality and communications patterns for parallel code.

2). Locality & Latency of Access Improvement

Can be grouped into 3 major categories:

- Compiler analysis for determining application memory access and communication behavior in single and multiprocessor systems.
- Cache design to improve locality of access and low-latency access.

- To reduce access latency by assigning pages to cache banks within the tile containing the core that touches the page first.

- Interconnect design to improve low-latency and high throughput communication.

- Create configurable networks to improve the communication latency for different types of communication patterns.

G. Compiler Analysis in data access

In the PARADIGM compiler, data usage in parallelizable Fortran 77 and HPF (high performance Fortran) code was analyzed and partitioned across machines with distributed memories. Optimized communication operations were generated for the parallelized code.

H. Compilation Framework

An advanced compiler framework based on array access regions. They reduced the data exchange by finding a suitable iteration/data distribution but did not consider cache coherence. proposed a profiling based scheme to determine a fine-grain data access partitioning. Another technique of describes an approach to partition data objects among multicluster processors. Work, size and usage information of each object is collected using data flow analysis. Objects associated with the same computation are merged together to form a group. Groups of objects are then partitioned for the multi-cluster machine to balance workload and improve performance (figure 1).

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 2, February 2014

The compiler is extended to include a phase partitioning algorithm and a methodology to target a hybrid electronic packet switched and optically circuit switched interconnect. However, these efforts only focus on explicit communications in MPIbased programs, omitting the implicit communication patterns implied by a shared-memory cache system.

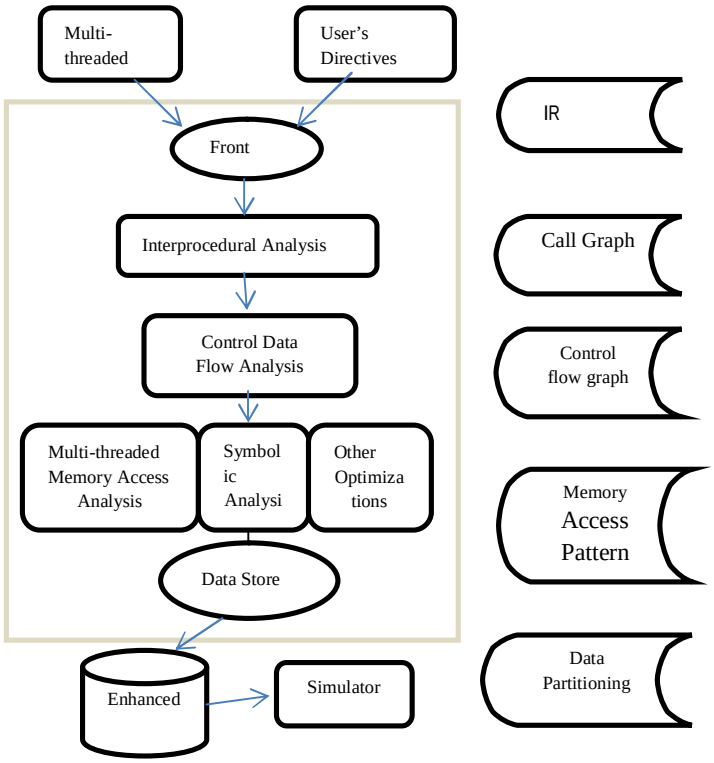

Figure 1. Compiler Framework

If array access appears N times in a m-deep nested loop, and the lower bounds, upper bounds, and steps of loop indices for thread x are □1(p),…□m(p), u1(p),… u m(p), and s1(p),… sm(p), respectively. Hence, the formula to calculate access weight for R(p)

$$W(R(p)) = N * \frac{M(U\square(p) - L\square(p))}{\sum_{K=1}^{m} S\square(p)}$$

To estimate the nonlocal access weight W, first need to compute the ending offset, overlap range and destiny of a region. Consider a region Ri(x) that has m span stride pairs. Sik(p) and Tik(p) are the Kth span stride within the region and O*i*(p) are the region's initial offset. Define the corresponding ending offset Ei(p) as the last element in the region.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 2, February 2014

$$Ei(p) = Oi(p) + \square \; Sik(p).$$
$$K=1$$

Thus, the overlap range of Ri(p) and R*(q) is defined as

$$\square = min(Ei(p), E*(q)) - max(Oi(p), O*(q)))$$

Where O*(y), E*(y) are the initial offset and ending offset respectively, of R*(y) The region R*(q)'s density D*(q) is the access weight of region relative to the range relative to the range of that region. Thus, D*(q) is defined as

$$D*(q) = * \frac{1}{E*(q) - O*(q)} W(R*(q))$$

IV. CONCLUSION

The Distributed System is one of the most important and widely used forms of shared permanent storage. Architecture, naming, synchronization, availability, heterogeneity and support for databases will be key issues that are to be taken into consideration while designing the Distributed System. Distributed System has provide different type of architecture. Data Distribution are used to improvement data Access locality and latency.

REFERENCES

1. C. Kim, D. Burger, and S.W. Keckler, "Nonuniform Cache Architectures for Wire-Delay Dominated On-Chip Caches," IEEE Micro, vol. 23, no. 6, pp. 99-107, Nov./Dec. 2003.
2. C. Bienia, S. Kumar, J.P. Singh, and K. Li, "The Parsec Benchmark Suite: Characterization and Architectural Implications," Technical Report TR-811-08, Princeton Univ., Jan. 2008.
3. J.D. Balfour and W.J. Dally, "Design Tradeoffs for Tiled cmp On-Chip Networks," Proc. 20th Ann. Int'l Conf. Supercomputing (ICS),pp. 187-198, 2006.
4. Z. Li and P. Yew, "Efficient Interprocedural Analysis for Program Parallelization and Restructuring," Proc. SIGPLAN Symp. Parallel Programming: Experience with Applications, Languages and Systems, July 1988.
5. B. Creusillet and F. Irigoin, "Exact versus Approximate Array Region Analyses," Proc. Ninth Int'l Workshop Language and Compilers for Parallel Computing, Aug. 1996.
6. Chandramohan A. Thekkath, et al, "Frangipani: A scalable Distributed WSFile System", System Research Center, Digital Equipment Corporation, Palo Alto, CA, 1997.
7. Barbara Liskov, et al, "Replication in the Harp File System", Laboratory of Computer Science, MIT, Cambridge, A, 1991.
8. Tanenbaum, van Steen: Distributed Systems, Principles and Paradigms; Prentice Hall 2002 Web site: www.prenhall.com/tanenbaum.
9. Coulouris, Dollimore, Kindberg: Distributed Systems, Concepts and Design; Addison-Wesley 2005 Web site: www.cdk4.net.
10. Compiler-assisted Data Distribution and Network Configuration for Chip Multiprocessors Yong Li, Student Member, IEEE, AhmedAbousamra,Student Member, IEEE, Rami Melhem, Fellow, IEEE, and Alex K. Jones, Senior Member, IEEE.

BIOGRAPHY

Amruta Salunkhe received the B.E. degree in Computer Engineering from the North Maharashtra University, Maharashtra, India, in 2010, doing M.E. degree in Computer Engineering from the University of Pune, Pune, Maharashtra, India in 2013.