1. INTRODUCTION

A heterogeneous distributed computing system (HDCS) consists of a set of processors or nodes of varying computing power, connected by a high speed network. An efficient scheduling of the tasks of an application on the available processors is one of the key factors for achieving high performance. There are two types of task allocation schemes: static and dynamic. In static task allocation scheme, the decision about which tasks should be allocated to which processors is made prior to the actual start of that job execution. In
dynamic task allocation scheme, the decision about
the number of processors to be made available for
a job and the task assignment to these processors
is made during the job execution and the decision
can vary according to the existing conditions
(current loads, node failures, priorities, etc.) of
the network. In recent years, HDCS has emerged
as a popular platform to execute computationally
intensive applications with diverse computing
needs. The problem of mapping (including match-
ing and scheduling) tasks and communications
is a very important issue since an appropriate
mapping can truly exploit the parallelism of the
system thus achieving large speedup and high
efficiency (Braun & Siegel, 1998). It deals with
assigning (matching) each task to a machine and
ordering (scheduling) the execution of the tasks
on each machine in order to minimize some cost
function. The most common cost function is the
total schedule length. This policy has a very high
overhead leading to a degradation of the overall
system performance. The above situation can be
avoided by the use of simple and constant time
heuristics. Efficient application scheduling is
critical for achieving high performance in het-
erogeneous computing systems. These heuristics
are classified into a variety of categories such as
list scheduling algorithms, task duplication based
algorithms, clustering algorithms and guided ran-
don search methods. Task assignment is known to
be an NP-complete problem. The natural approach,
therefore, is to develop good heuristics that will
allocate the tasks to the available processors and
minimize the schedule length.

List based scheduling: According to their pri-
ority, list-scheduling heuristic maintains a list of
all tasks of a given graph. It has two phases: task
prioritizing phase for selecting the highest-priority
ready task and the processor selection phase for
selecting a suitable processor that minimizes a
predefined cost function. Some of the examples
are the Modified critical Path (MCP) (Wu &
Gajski, 1990), Dynamic level Scheduling (Sih
& Lee, 1993), Mapping Heuristic (MH) (Rewini
& Lewis, 1990), Insertion Scheduling Heuristic
(Kruatrachue & Lewis, 1998), Earliest Time
first (EFT) (Hwang, Chow & Anger, 1989), and
Dynamic Critical Path (DCP) (Kwok & Ahmad,
1996) algorithms. List-scheduling heuristics are
generally more practical and provide better per-
formance results at a lower scheduling time than
the other groups.

Task Duplication based Heuristics: The idea
behind duplication-based scheduling algorithms
is to schedule a task graph by mapping some of
its tasks redundantly, which reduces the commu-
nication overhead (Ahmad & Kwok, 1994; Park,
The only change is to selection strategies of the
tasks for duplication in different duplication-based
algorithms.

Clustering based heuristics: Another class of
DAG scheduling algorithms is based on a tech-
nique called clustering (Gerasoulis & Yang, 1992;
Kim & Browne, 1998; Yang & Gerasoulis, 1994).
The basic idea of clustering based algorithm is to
group heavily communicated tasks into the same
cluster. Tasks grouped into the same cluster are
assigned to the same processor in an effort to
avoid communication costs.

Guided random search algorithms: The task
scheduling problem is a search problem where the
search space consists of an exponential number
of possible schedules with respect to the problem
size. Guided random search algorithms are a class
of search algorithms based on enumerative tech-
niques with additional information used to guide
the search. They have been used extensively to
solve very complex problems. A Genetic algo-
rithm (GA) (Davis, 1991) is a type of evolution
computations that is commonly used.

2. TASK-SCHEDULING PROBLEM

An application program can be represented by a
Directed Acyclic Graph (DAG), G = (V, E), where
V is the set of n tasks. E is the set of e directed edges.