A Clustering Approach for Task Assignment Problem

Vinay Kumar
School of Computer and System Sciences
Jawaharlal Nehru University
New Delhi, INDIA

P. C. Saxena, PhD.
School of Computer and System Sciences
Jawaharlal Nehru University
New Delhi, INDIA

C. P. Katti, PhD.
School of Computer and System Sciences
Jawaharlal Nehru University
New Delhi, INDIA

ABSTRACT
The problem of task assignment in heterogeneous computing system has been studied for many years with many versions. We consider the problem in which tasks are to be assigned to homogeneous and heterogeneous machines to minimize the sum of the total computation and communication costs. In this paper, we introduce a novel algorithm to solve task assignment problem. It optimizes the assignment of cluster according to the storage and load balancing constraints and converts task assignment problem into a linear programming problem under the constraints of memory availability and load balancing on each machine. The aim of this work is to increase throughput, maximize resource utilization and fulfillment of user requirements.

General Terms
Distributed Computing, Heterogeneous Computing

Keywords
TaskAssignment Problem;Task Scheduling; Load Balancing.

1. INTRODUCTION
In recent years, work of network technology has made distributed computing system an attractive alternative to massively parallel machines [1]. To exploit the capability of these systems for an effective parallelism, the tasks of an application must be properly assigned to the machines. A heterogeneous computing environment that consists of a heterogeneous suite of machines and high-speed interconnections provides a variety of architectural capabilities, which perform an application that has diverse computational requirements [2, 3, 4, and 5]. The performance criterion for heterogeneous computing used in this paper is to minimize the completion time, i.e., the overall execution time of the application on the machine suite. Partitioning of tasks and assigningto machines is an important issue in homogeneous and heterogeneous environment. The problem of task assignment in homogeneous and heterogeneous system deals with finding proper assignment of tasks to machine in order to optimize some performance metric such as the system utilization and turnaround time. Task partitioning and assigning problem are discussed in many papers e.g. [6, 7, 8, 9, 10].

In min-min heuristic [6], minimum completion time for each unmapped tasks respect to all machines is calculated. A task is then selected that has overall minimum computational time and assigned to corresponding machines. The unmapped tasks set U are updated and the process is repeated until all tasks are mapped. Max-min [6] is very similar to min-min except that a task is selected that has overall maximum computation time instead of minimum computation time. Again U is updated and process is repeated until U is empty. The rationale behind suffrage [7] is that a task should be assigning a certain machine that would "suffer" the most if not assigned to that machine. For each task, its sufferage value is defined as the difference between its best MCT (minimum computational time) and second best MCT. Tasks with high sufferage value take precedence. An improvement in suffrage heuristic is Xsurffrage [8]. In this task suffrage value define on cluster level not on task level to each task. It removes the chance of late assignment of a task in suffrage heuristics.

It is well known that the problem of partitioning and assigning of tasks is NP-Complete in general [11] and capacity of machines forms an important part of scheduling algorithm. In the given literature only minimization of communication cost is considered. In this work, minimization of computational cost is also considered with communication cost. We study task assignment problem as an optimization problem with two constraints one for storage that is available on machines and other is load balancing for maximize utilization of machines.

2. ALGORITHM
Consider a job *T* that divided into *m* tasks t_i (*i=1,2,3....m*) and job *T* executed on *n* machines u_k (*k=1,2,3,...n*). Suppose the running cost of t_i on u_k is denoted by R_{ik} *(1 ≤ i ≤ m, 1 ≤ k ≤ n)* and is known a priori. If any taskt_i not executable on u_k , then running cost R_{ik} is taken as infinite. The communication cost between $t_i$and t_j is denoted by C_{ij} *(1 ≤ i, j ≤ m)*. Thus C_{ij} is incurred only when t_i and t_j are executed on different machines. Each machine has a fixed amount *S* of storage available for the task in *T*. The storage requirement of task t_i is represented by $S(t_i)$ and is known as priori. Assume that total storage demand by job *T* is less than the storage capacity of system.

An optimal assignment of task t_i's to machines must minimize the total cost of job *T* comprising of the running cost of tasks. Further assignment must be such that the storage requirement on a machine is less than the total capacity *S*. Firstly the task assignment problem on homogeneous machines is discussed then the extended for heterogeneous machines. Suppose all machines are homogeneous i.e. they have same computing power. The cost that can be minimized is communication cost only in this system, running cost will be same for job *T*.

2.1 Algorithm for Task Assignment Problem
Construct the *n* disjoint cluster by partitioning of *m* tasks, where each cluster has been associated to a distinct machine. Let partitions of tasks $P=(P_1, P_2, P_3, \ldots..)$, where P_1, P_2, P_3denote the cluster of tasks. The formation of tasks assignment problem express as

Minimize $\sum C_{ij}$ [$t_i$and t_j are in separate cluster]
Overall partition $P= (P_1, P_2, P_3, \ldots\ldots P_n)$

Subject to
i) Storage $S(P_r) \leq S$, $\forall r=1,2,3....n$(1)
ii) Load Balancing $L(P_r) \leq L$, $\forall r=1,2,3....n$

Where$S(P_r)$ and $L(P_r)$ are storage and computational load on cluster P_r respectively.

In our proposed algorithm we give a procedure to generate all feasible partition satisfying constraint *i)* and *ii).* Let Z_a ($a=1,2,3....m$) denotes the set of all partitions of tasks t_1 , t_2 , t_3 ……t_a. Clearly $Z_1 = (t_1)$ is containing the partition that have only one cluster made-up by t_1. To generate Z_m, the set of all partitions of t_1 , t_2 , t_3 ……t_m, we use an iterative procedure *PART*, which derives Z_a from $Z_{a-1}$for *a=2,3,4....m-1*.
In procedure *PART*, let $P=(P_1, P_2, P_3, \ldots . P_r) \in Z_{a-1}$. Generate *r* new partitions to belong to Z_a from *P*, by adding t_a to one cluster of *P* at a time. If *r* is less than total number of machines, then generate another partition $P' = (P_1, P_2, P_3, \ldots \ldots P_r, P_{r+1})$, where $P_{r+1} = (t_a)$ and include it in Z_a. Apply above steps for all clusters of Z_{a-1}.

PART(Z_{a-1}, Z_a)
1 begin$Z_a \leftarrow \Phi$
2 for each partition$P \in Z_{a-1}$*do*
3 begin let P be $(P_1, P_2, P_3, \ldots \ldots P_r)$
4for i←1 to r do
5 begin$P' = (P_1, P_2, P_3, \ldots \ldots, P_i \cup t_a, \ldots \ldots P_r)$,
*6*$Z_a \leftarrow Z_a \cup P'$
7end;
8 if r<n then
9 begin$P' = (P_1, P_2, P_3, \ldots \ldots P_r, t_a)$
*10*$Z_a \leftarrow Z_a \cup P'$
11end if;
12 end;
13 end of PART

An example of enumeration of all partition is given in Fig 1, where a job is divided in *4* tasks and system has *2* machines. The nodes in figure correspond to partitions. The collection of nodes at *(a-1)th* level corresponds to set Z_a. A node is called infeasible in the tree if any of the clusters in corresponding partition violates the constraints *i)* and *ii).* By the property of this tree we conclude as in next theorem.

Theorem2.1- If a node *P* in tree is infeasible, and then all nodes of the subtree with root *P* are also infeasible.
Proof - Suppose $P = (P_1, P_2, P_3, \ldots \ldots P_e)$ is infeasible, so we must have either $S(P_i) > S$ *or* $L(P_i) > L$ for at least one *i*, $1 \leq i \leq e$. Consider an arbitrary node $P' = (P'_1, P'_2, P'_3, \ldots \ldots P'_f)$ of the subtree with root *P*. By definition we must have for $f \geq e$, $S(P'_r) \geq S(P_r)$ and $L(P'_r) \geq L(P_r)$ $\forall r=1,2,3....e.$ Thus if $S(P_i) > S$ then $S(P'_i) > S$ and if $L(P_i) > L$ then $L(P'_i) > L$, showing P' is infeasible.
We will use this result in generating only feasible partitions. In *PART*, nodes are generated from root to leaf, and generate all possible partitions. In next algorithm, when we detect an infeasible node during generation of tree, this node may be deleted, thereby eliminating the subtree to be generated from it. The next algorithm gives Z_m (set of all feasible assignment) for our problem for homogeneous machines. In this the running cost R_{ik} *(*$1 \leq k \leq n$*)* taking as R_i.

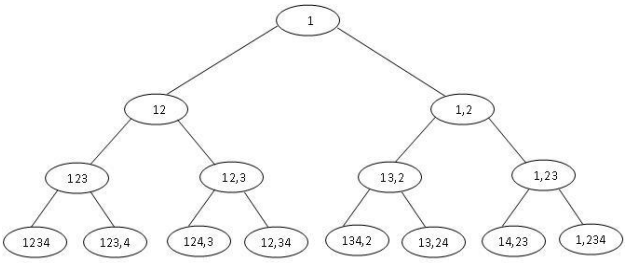

Fig 1: Enumeration of all Partitions of 4 tasks and 2 machines

PROC
1 begin p← t_1*; S(P) ←* $S(t_1)$
2 $L(P) \leftarrow R_1$*;* $Z_1 \leftarrow [P]$
3 for a← 2 to m, do
4 begin$Z_a \leftarrow \Phi$
5 for each partition$P \in Z_{a-1}$*do*
6 begin let P be $(P_1, P_2, P_3, \ldots \ldots P_r)$
7for i←1 to r do
8 begin $P'_i \leftarrow P_i \cup t_a$*;* $S(P'_i) \leftarrow S(P_i) + S(t_a)$; $L(P'_i) \leftarrow L(P_i) + R_a$ *;*
9 if $S(P'_i) \leq S$ *and* $L(P'_i) \leq L$*, then*
10 begin$P' \leftarrow (P_1, P_2, P_3, \ldots \ldots, P'_i, \ldots \ldots P_r)$,
*11*$Z_a \leftarrow Z_a \cup P'$
12end;
13 if r<n then
14 begin$P' = (P_1, P_2, P_3, \ldots \ldots P_r, t_a)$
*15*$Z_a \leftarrow Z_a \cup P'$
16 end;
17 end;
18 end;
19 end;
end PROC

An example of performing *PROC* is given in Fig2 and 3. Suppose $P_{3,3}$(node *3* on depth *3*) does not satisfy storage constrain *i).* Thus $S(12) \leq S$ but $S(124) > S$, where $S(124) = S(12 \cup 4)$. For this case *PROC* code is not executed from line *9* to line *12*, as shown in Fig*2*. Further assume that $P_{2,3}$(node *3* on depth *2*) is violating the load balancing constraint ii). Thus $L(1) \leq L$ but $L(13) > L$, where $L(13) = L(1) + R_3$. Statement in line *9* is not true, thus code is not executed from line *9* to *12.* The output is given in Fig 3

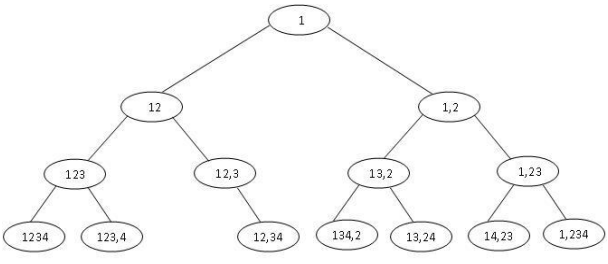

Fig 2: when storage constraint not satisfied

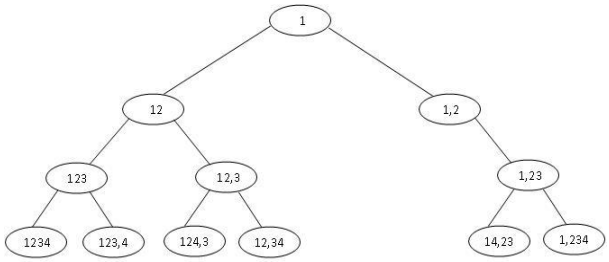

Fig 3: when load balancing constraint not satisfied

Another example is given below for performing *PROC* with *5* tasks and *2* machines in Fig 4. In this figure enumeration of all partition at last level can easily build by adding task *5*. Let $P_{3,4}$(node *4* on depth *3*) does not satisfy storage constrain *i).* Thus $S(3) \leq S$ but $S(34) > S$, where $S(34)=S(3 \cup 4)$, as shown in Fig 5. Further assume that $P_{2,4}$(node *4* on depth *2*) is violating the load balancing constraint ii). Thus $L(2) \leq L$ but $L(23) > L$, where $L(23) = L(2) + R_3$. The output is given in Fig 6.

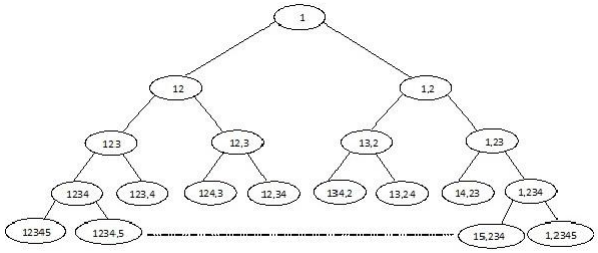

Fig 4: Enumeration of all Partitions of 5 tasks and 2 machines

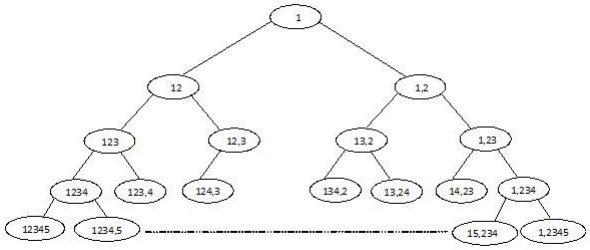

Fig 5: when storage constraint not satisfied

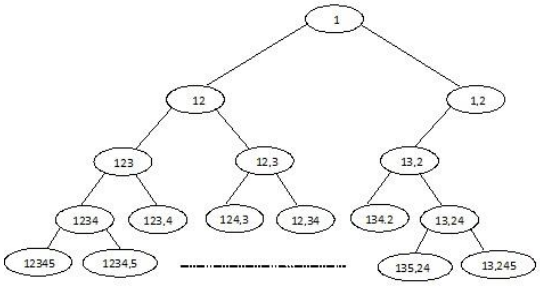

Fig 6: when load balancing constraint not satisfied

In other case, when all machines are heterogeneous then they do not have the same computing speed. In this case, cost function is to be optimized by including running costs along with the communication cost between machines. Some different types of solutions of Task Assignment Problem for heterogeneous environment are discussed in recent years [12, 13]. A hybrid meta-heuristic approach for heterogeneity version of this problem is also presented by Sanz, Yao and Xu [14].

The Task Assignment problem for heterogeneous environment is stated as:
Minimize $\beta_r + \beta_c$, where β_r is the running cost and β_c is the communication cost over all feasible assignment subject to storage constraints of equation (1).

This problem for heterogeneous machines is NP-Complete in general. We use results of previous problem to solve this optimization problem. In the case of homogeneous machines, the partition of tasks corresponds to a single assignment, but in this we take partition of tasks corresponds to a well-defined subset of assignments. Let *P* denote a partition of tasks in *T* and subset P^*denote the collection of assignment corresponding to *P*. Then
$P^* = \{P^i : P^i$ is an assignment derived by associating different machines to cluster in $P\}$
All elements of the set $P^*$are either feasible or infeasible. The communication cost between machines is same for each element P^i, because no two clusters are associated to the same machine. If the earlier proposed algorithm applied here, then the collection of all feasibleassignment will be in the form of disjoint subsetsP^*. The optimal assignment from this collection could be determined in two stages
a) Determine the best assignment in each subsetP^*.
b) Compare all such locally best assignments to find out optimal assignment.

First stage may be formulated as the well-known classical assignment problem as
Let $P = (P_1, P_2, P_3, \ldots\ldots P_e)$, consider a cluster $P_j \in P$. Define a variable Y_{jk} as follows
$Y_{jk} = 1$, if P_j is associated to machine u_k
$Y_{jk} = 0$, otherwise
Let *Y* be the vector of Y_{jk} , $1 \leq j \leq e,\ 1 \leq k \leq n$
The problem is to find an assignment in P^*, which has a minimum running cost. Running cost represent by *RC* as

$$RC(Y) = \sum_{j=1}^{e} \sum_{k=1}^{n} Y_{jk} \{\sum_{z_a \in p_j} R_{ak}\}$$

Thus problem can be express as
Minimize *RC(Y)*
Subject to
i) $\sum_{j=1}^{e} Y_{jk} \leq 1, \forall k = 1,2,3\ldots\ldots n.$
ii) $\sum_{k=1}^{n} Y_{jk} = 1, \forall j = 1,2,3\ldots\ldots e.$
iii) $Y_{jk} = 0$ or $1, \quad \forall k = 1,2,3\ldots n$ and $\forall j = 1,2,3\ldots\ldots e.$

Constraint *i)* tells that at most one cluster is associated to a machine. Constraint *ii)* ensures that each cluster is associated to one and only one machine. This linear integer programming problem can be solve by 0-1 technique like branch and bound method [15, 16]. The complexity of homogeneous and heterogeneous machines case depends upon the stringency of constraints that are taken in optimization problem.

3. CONCLUSION

In this paper we have proposed a simple model for analysis of task assignment problem in homogeneous and heterogeneous environment. We consider memory availability and load balancing on each machine to solve task assignment problem. The objective considered in proposed algorithm is to minimize computational cost of tasks and communication cost between machines. Firstly, algorithm has been discussed for homogeneous machines then for heterogeneous machines. The general case is known be NP-Complete, however, the proposed system here could be a benchmark for evolution of other heuristics algorithms. Future work can be extended by considering other constraints like data availability on machines, dependency of tasks, different bandwidth between machines etc.

4. ACKNOWLEDGMENTS

This research is supported by Capacity Buildup Fund, JNU, New Delhi, INDIA

5. REFERENCES

[1] Foster I and Kesselman C (editors), (1999), The Grid: Blueprint for a Future Computing Infrastructure, Morgan Kaufmann Publishers, USA.

[2] M. M. Eshaghian, ed., Heterogeneous Computing, Artech House, Norwood, MA, 1996.

[3] A. Khokhar, V. K. Prasanna, M. Shaaban, and C. L Wang, ''Heterogeneous computing: Challenges and opportunities,'' IEEE Computer, Vol. 26, No. 6, June 1993, pp. 18-27.

[4] H. J. Siegel, J. K. Antonio, R. C. Metzger, M. Tan, and Y. A. Li, ''Heterogeneous computing,'' in Parallel and Distributed Computing Handbook, A. Y. Zomaya, ed., McGraw-Hill, New York, NY, 1996, pp. 725-761.

[5] H. J. Siegel, H. G. Dietz, and J. K. Antonio, ''Software support for heterogeneous computing,'' in The Computer Science and Engineering Handbook, A. B. Tucker, Jr., ed., CRC Press, Boca Raton, FL, 1997, pp. 1886-1909.

[6] Braun R, Siegel H, Beck N, Boloni L, Maheswaran M, Reuther A, Robertson J, Theys M, Yao B, Hensgen D and Freund R, (2001), A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems, International Journal of Parallel and Distributed Computing, Vol.61(6): 810-837.

[7] Maheswaran M, Ali S, Siegel H.J, Hensgen D. and Freund R. F,(1999), Dynamic Matching and Scheduling of a Class of Independent Tasks onto Heterogeneous Computing Systems, International Journal of Parallel and Distributed Computing,Vol. 59(2):107-131.

[8] Casanova H, Legrand A, Zagorodnov D and Berman F,(2000), Heuristics for Scheduling Parameter Sweep Applications in Grid Environments, In. Proc. Of the 9th heterogeneous Computing Workshop**:** 349-363, Cancun,Mexico.

[9] Stone H S, (1977), Multiprocessor Scheduling with the aid of network flow algorithms, IEEE Trans. Software Eng.3:85-93.

[10] Stone H S, Bukhara S H,(1978), Control of distributed Processes, Computer: 97-106.

[11] Rewini H. E, Lewis T, and Ali H, (1994), Task Scheduling in Parallel and Distributed Systems, ISBN: 0130992356, PTR Prentice Hall.

[12] Hinma Kang, Hong He and Hui-Min Song, "Task Assignment in hetrogeneous computing systems using an effective iterated greedy algorithm",Journal of System and Software 84(6), pp 985-992, 2011.

[13] Peng-Yeng Yin, Yung-Pin Cheng and Benjamin B.M. Shao, "Metaheuristic algorithms for Task Assignment in distributed computing system: A comparative and integrative approach", The Open Artificial Intelligence Journal, pp. 16-26(11), 2009.

[14] S. Salcedo-Sanz, X. Yao and Y.Xu, "Hybrid meta-heuristic algorithms for Task Assignment in heterogeneous computer systems", Computers & Operations Research, vol. 33, no. 3, pp. 820-835, 2006.

[15] I. S. Hillier and G. J. Lieberman, "Introduction to Operations Research (4th Ed)", CBS publications and distributors, 1985.

[16] H. A. Taha, "Operations Research: An Introduction", Prentice Hall Inc, New Jersey, 1997.