Science Arena Publications
Specialty Journal of Electronic and Computer Sciences
Available online at www.sciarena.com
2017, Vol, 3 (2): 1-9

A Review Based On Scheduler Algorithms

Pouriya Saeedaskari[1*]
[1]M.sc., Department of Information Technology, Faculty of Sciences, Kerman branch, Islamic Azad University, Kerman, Iran.*
E-mail: pouriyalpp@gmail.com tell:09336429179

Abstract: *Remarkable progresses and information technology's increased, deep and effective changes in the business environment are going on. Super-computing is one of the most innovative distribution models and IT services that can be named as the development and deployment of Internet-based services or super-computing. Super-computing is a model of distributed computing, consisting of a large number of resources and applications with the goal of sharing resources as a service over the Internet, and more efficient use of resources is a timeless challenge, hence the problem of scheduling tasks in super-computing, is considered as important that seeks an optimal scheduling to execute tasks and determine the optimal resource allocation. In this regard, different algorithms with different features there exist that are discussed in this article.*

Keywords: scheduling, super-computing, statics algorithms, real-time algorithms, workflow algorithms

INTRODUCTION

Today IT and internet are an integral element of people's lives. With the changing lifestyle, needs of society such as information security, faster processing, more immediate access to information and the most important of them cost savings have been changed. Consequently, by means of developing these needs, organizations and individuals have quite different needs in the field of electronic services, respect to the past. Super-computing is a developed technology which enables the IT industry to reduce computational costs. In a super-computing environment, task scheduling and resource allocation, service providers are managed through virtualization technology. They are used for hiding and completing user tasks in a transparent manner. Task scheduling becomes even more complex because of the transparency and flexibility of super-computing and different needs for resources. Task scheduler based on justice or resource efficiency strategies focus on that the cost of time, space and operational power increase and quality of services will improve in super-computing. (Hong et al., 2013). Workflow means a set of tasks with dependencies between them. In the workflow model, a process consists of simple steps that reduce design complexity and management needs of an application program. Recent advances in virtualization technologies and rapid grow of super services in recent years, cause to create a new paradigm for distributed computing, on the super, in order to use existing and scalable resources. The Business Process Management System has been created in order to adapt to a new paradigm and to take advantage of super services. In this regard, the platform as a service provides an execution environment so as to create and deploy workflows in super infrastructure (Du & Li, 2008). Workflow scheduling process refers to the mapping task (a group of tasks) to the available computing resources and scheduling their run time so that dependency between them to be preserved. Workflows structures often are defined as a directed acyclic graph (DAG) and more like to a tree structure (Yu & Buyya, 2005).

Scheduling tasks algorithms:

The traditional methods which are used in optimization, are deterministic and fast and usually reply accurately, but often act locally. Task scheduling is more difficult in the large search space and has a large inventory of solutions and tasks must spend more time for finding the optimal solution. To solve this problem, in this situation, the desired method is not defined. In Continue, we will discuss a variety of static scheduling algorithms, including static, real -time and workflow algorithm.

Static and dynamic models for scheduling tasks

In super-computing, a typical data center is composed of high-speed machines. This medium calculates a large and diverse group of tasks. Tasks to different users are distinguished from each other. In such circumstances, the execution of multitasking scheduling relates to several machines. In general, exploration mode as a semi-optimal algorithm is supposed to be good solutions. Among them are static and dynamic algorithms (Armstrong et al., 1998). In the following, they have been described and some algorithms which possess the same scenarios have been reported.

Static strategy:

The static heuristic method is used when the full set of tasks to run are recognized. These strategies are based on two assumptions. The first is, simultaneous arriving of the task and the second assumption is that the available machines time, being updated after each scheduling.

Dynamic strategy:

The dynamic heuristic method is necessary when machines and set of tasks are not fixed. For example, all tasks do not arrive at the same time or any of the machines fail or go offline from time to time. The dynamic heuristic method is used in two modes, online and batch. In the first case, when a task arrives at a machine, this machine is scheduled and each task is scheduled only once for each timing outcome cannot be changed. The heuristic online mode is appropriate for low input rate. In batch mode, first, the tasks are collected and scheduling is fulfilled in a series of pre-planned times and in this type of scheduling, we can be aware of accurate number of scheduling.

Opportunistic Load Balancing:

The basic idea of opportunistic load balancing algorithms is keeping all processors busy as much as possible. In this algorithm, no matter when performing tasks on resources, regularly all resources will be checked and, assigning tasks to resources will be done. The algorithm will assign the task to corresponding resources after source work is done, and there exist task which is not scheduled. Indeed, when tasks reach to the system resource management and stay in turn and any source to release soon, the head of the queue is assigned to the task. This algorithm attempts to keep all sources busy and use them up to the maximum rate and distribute the load to the same amount. One advantage of OLB is its simplicity, but since that, the algorithm never takes into consideration the execution time, typically it has got high completion time. In contrast to OLB algorithms, we have MET and MCT algorithms, with an aim to reduce time execution and task completion time (Freund et al., 1998).

Minimum Execution Time algorithm:

The main idea of the minimum completion algorithm is combining the advantages of both OLB and MET methods. MCT algorithm allocates a task to a processor with less time to complete that. This results in some

assignment of tasks to processors that their execution time is not good. Once a task enters to a scheduler, all processors in the system are examined and the processor which has the lowest completion time for the task is determined (Braun et al., 2001).

Min-Min Algorithm

This approach is prioritizing tasks and scheduling production based on priority. This priority is generated based on expected task completion time. This method adjusts the tasks in several independent tasks. Then these groups are then repeatedly scheduled. Each iteration takes a set of unmapped independent tasks and creates minimum expected completion time (MECT) for each task. A task that owns a smaller amount of MECT, more than all the selected tasks to the appropriate resources, in the first iteration is scheduled. This continues until all tasks are scheduled (Maheswaran et al., 1999). The purpose of this algorithm is to achieve the least response and for aim this, algorithm schedules the tasks with less time and then with more time in ascending order.

Min-Max algorithm

This method is similar to the Min-Min method. Min-Max algorithm starts with a set of all non-scheduled tasks and computes the minimum completion time in the set for each task. This algorithm's discrepancy with previous methods is that the tasks with most completion time are selected and mapped to machine. This process continues until all tasks to be scheduled.

Genetic Algorithm

The algorithm is a set of evolutionary techniques derived from nature that has lots of applications in optimization problems and being employed in searching in large spaces. Initially, for scheduling a specific task, an initial population of chromosomes is randomly generated. Each chromosome has a ratio, here this ratio is the whole execution time corresponding to a chromosome. After creating the initial population, for all the chromosomes in the population, the ratio will be assessed and the chromosomes which have the lowest proportion will be the best mapping. The algorithm then enters a loop consists of four phases: selection, crossover, mutation, and assessment. In the selection phase, some chromosomes are replicated and some of them are removed so that better maps which have more probability cloud be duplicated in the next generation. Then the features which ensure the survival of solutions will be implemented in population and the population size remains constant. In the reproduction, a chromosome pair is selected and a random point in the first chromosome picked out. After this operation means reproduction, mutation operation is run. The genetic mutation selects a chromosome and then picks a task in it and assign to a new machine. At this point, all three acts are done randomly. Finally, the chromosomes of the changed population are re-evaluated. This process continues until stop conditions meet (Braun et al., 2001).

Simulated Annealing algorithm

Simulated annealing Algorithm, is a repetitive technique which examines just a mapping solution for each task at a time. This solution uses the same chromosome representation in genetic algorithms. This algorithm picked out a method based on probabilities give permission and tries to have the best search in solution space. This probability is based on system temperature which decreases in each cycle. Cooling of the system will make harder the process of acceptance of weaker solutions. In the implementation of this algorithm, the initial mapping method will be produced by a uniform random distribution. The mapping will be mutated the same way in genetic algorithms, overall execution time will be assessed. If the new overall execution is better,

new maps will replace the old. If the overall execution time is longer and not bad, a uniform random number is selected and the examination is done according to this phase of the algorithm (Zumaya & Kazman, 2010).

Tabu algorithm

Tabu search is a solution space which writes some regions of the solution that has been searched former up to next search nearby of this region has not been searched. One solution in mapping, is using the chromosome representation in the genetic algorithm. Implementation of tabu search starts by generating the random mapping from a uniform as an early solution. To manipulate the current solution and move in the solution, a short step will be covered. The aim of the short step is to find the nearest local minimum solution in the solution space (De Falco et al., 1994).

A* algorithm

A heuristic search method is based on the starting tree and initializes from a root node which is null. As the tree grows, nodes represent the partial scheduling (a subset of tasks dedicated machines) and leaf represents the final scheduling (all tasks dedicated to machines). A partial solution has got a child node and a task from father node. Each parent node can be replaced by their children. To keep the heuristic execution time, a pruning process, there exists in order to limit the maximum number of nodes in the tree at any time. This method is quite equivalent to a perfect search. The tree is not pruned, this method of search is complete. This process continues until the schedule is complete (Braun et al., 2001).

K-Percent Best algorithm

The algorithm only considers a subset of processors for allocation. A task which has the closest completion time in the preceding subset will be allocated to the processor. If K = 100, then the algorithm is reduced to the MTC.

Min-Max algorithm

The algorithm works in a way that at every step of the tasks and the set of processors, picks out a set of paired task processor, which has got the closest expected completion time; but unlike the previous, this algorithm selects a set of paired task processor that has got the most expected completion time and allocates the preceding task to the processor. In the case, that number of short tasks are more than long tasks, it is expected this algorithm will perform the same process better than previous algorithms.

Suffrage algorithm

This algorithm is a heuristic method and each task will be assigned to source, based on the amount of suffrage. The amount of suffrage will be defined as a difference between the first and second completion. A task with large expected will be assigned to a machine with least completion time (Maheswaran et al., 1999).

Workflow scheduling algorithms

The workflow scheduling is one of the key issues in the management of workflow, especially in the grid and super workflow systems (Radulescu & van Gemund, 1999). This is a process that manages the execution of tasks related to distributed resources. The allocation of adequate resources for workflow tasks can be completed for satisfactory of objective functions by the user. Here are some of the algorithms described in this area:

Fast Critical Path Algorithm

Fast critical path algorithm, attempts to reduce the complexity of task scheduling while this algorithm preserves the functionality of scheduling and this complexity can be decreased effectively if available arranged task has constant size. In this phase, it will be saved in a simple unordered FIFO list that has gotten O (1) access time. When a task is ready, if there is a place for it to be added then it is listed, otherwise, it will merge in the FIFO. Tasks are always removed from the arranged list, then, if the FIFO list is not empty, a task will move to the list. The complexity of the sorting task time reduces using a list of size H to O (HlogH), and all tasks only arranged once and are picked up. The possible weak point is the use of the fixed-size list that it is possible the task with the highest priority could not be in the ordered list, but temporarily is saved in the list of FIFO. So the list should be large enough to have an algorithm with high performance. At the same time, it should not be too much complex. Author experiments show that a priority list with size "p" is a good choice for FCP algorithm.The smaller size with limited parallelism problems, while large size does not mean more performance improvements (Radulescu & van Gemund, 1999).

Adaptive Generalized Scheduler algorithm

In the adaptive generalized scheduler algorithm, one of the important classes is file store important, which has been implemented file as a vector of the time gap. So when we talk about all computation nodes for scheduling tasks or other tasks which are assigned to nodes, gap time vector will include information earlier than a cycle. First, for each task, the start time is determined by the end of the last parent process. Next, the computing node in which the last parent was processed, is determined. Data transmission delay is calculated from other parents of the node and starting time is renewed. If a node which has been updated since the beginning up to the end of to task processing, were available, the task is assigned to the node. Otherwise, start time will be calculated for each after the assessment of the evaluation process. Using the updated start time, the end time of the process for the node can be found. All available nodes task will be mapped to the computational node that has the closer end time. After mapping, gap time will be updated (Agroval & Kent, 2005).

Workflow Mapping Mechanism

The main purpose of workflow mapping mechanism is to find an optimum choice in respect to user recalled service quality criterion and to present by service providers. The major stages of the algorithm are summarized in the following:

- Computing characteristic values for algorithm completion and defining the criterion which describe QOS presented level by means of services.
- Mapping workflow onto a service sample initializes without exceeding the cost limits, respect to user need for access purposes.
- It defines candidate service sample for every type of services. The reason is exploring candidate service quality for each kind of service in order to present high quality level of services. For finding feasible alternatives, it creates a list with the best candidates.
- An outline gives permission for picking more than one service sample out in each type of services and if the user does not face with a deadline.

Adaptive Workflow Splitting Algorithm

In adaptive workflow splitting algorithm, when a scheduler receives a task, it starts computing the communication to computation ratio for DAG graph and number of source cluster for example source N. The higher value for CCR means that task graph is computational compressed. Then, the scheduler chooses N

source clusters with high score respect to their knowledge. An iteration graph examines the partition of each residual node in G in order to determine if this node can be placed into a subgraph like a G'. It will look for some edge with high communication that has not been examined. If we have this kind of edge, this means that all of the nodes which have not been checked right now will be merged in G', scaling of first depth of graph from two vertices is going to lead merging of all task nodes. The aim of graph scaling, is that task without exceeding the source threshold limit, they merge into a subgraph and at the same time, for preventing unnecessary breaks, graph scaling preserves the priority orders of task that helps prevent parallelism increase but it may raise the communication cost between the clusters. A stack is used for manipulating the node task. If this edge could not be merged in G', the responsible node will be determined. This means a new subgraph is going to be generated. The Scheduler repeats this loop until all nodes in G can be allocated. After algorithm completion, each scheduler distributes every subgraph to certain source cluster (Dong & Akl 2007).

Loss and Gain Approach

Loss and gain approach, adjusts the scheduling which has been generated from a cost perspective by means of optimum heuristic time or optimized heuristic to cope with budget constraints. Scheduling starts by initial assignment of task into machines, and computes each task variation into a different machine on the basis of related weight to the specific change. These values' weight, will be illustrated regularly in a table, thus a weight table for each task and each machine will be created. Two alternative methods have been proposed for computing weighted values in regards to two selects for initial assignment. Owning the optimum completion time or optimum cost, by the way algorithm tries up transmission each pair of tasks not to happen more than once and when no transmission happens, the algorithm terminates (Sakellariou et al., 2007).

Real time scheduling algorithm

There exists some group of application which take care of the time. About these programs which have critical deadline, even a small delay will help system to crash. For example, the traffic control centers gather information about the latest status of the roads and save them into a data base. If a user sends a request for last status to the database, a real time decision will return proper control information by the operators. But the last agreement of super service, does not provide users with real time control underneath of time behaviors, thus a clear, flexible and more reliable agreement will be needed. Priority for these kind of tasks will be used on the basis of deadline. The scheduler will give a task to source based on priority policy. Based on this, scheduler priority will split into two groups: Static priority strategies and dynamic priority strategies.

Statics priority strategy:

One real time task consists of all samples. In scheduling of all same and constant priorities, all samples have got the same and equal priority. The most powerful algorithm is the rate monotonic. In this algorithm, the priority of a specific task associates with its freedom degree. If this criterion is high, the priority corresponding to this task is too high (Liu & Layland, 1973). In this algorithm, the deadline of the task was not equal to its period. The priority was not optimum for allocating and it was considered as a weak algorithm. Therefore, the monotonic algorithm will assign a high priority to the task with short deadlines, respect to the task with the long ones (Lenng & Whitehead, 1982).

Dynamic priority strategy:

The dynamic assignment method is more efficient than the constant behavior, since this strategy can take advantage of processing from all components of processors. The priorities are changing continuously in time

and from one request to the other one. The most used algorithms in these models, are earliest deadline and least laxity. The earliest deadline algorithm, give priorities to tasks in an inverse order and proportional to the deadlines of active operations. The least laxity algorithm assigns the processor to the active task corresponding with least laxity and this laxity may alter several times.

Real time scheduler

A dynamic scheduler, makes scheduling decisions at the execution time and maybe selects the task out of the current tasks. The static scheduler makes decisions before execution time. In general, the real time schedulers have been embedded into cores, since their scheduling approach. The aim of the Mars core in hard real time systems, is reaching to high loading status. It has been consistent with scheduling approach. The scheduling will be computed offline and it feeds the nodes with some part of system inputs. The ART core will help us in providing a predictable distributive system which is analyzable and reliable. This core will use LIF, EDF and RM algorithms for analyzing and ensuring offline hard real time processes. Nonperiodic hard real time processes will be scheduled by using a deferrable server. All other processes are scheduled by means of operational value and in dynamic form. (Tokuda & Mercer 1989). Increasing real time services, real time cores will be needed more in super mediums. However, a large number of cores there exist, they are not able to meet real time system needs, specially system with multitude cores. Regardless of core configuration, placing a real time scheduler in operating systems will be considered as a solution. A simple example is RESCH for Linux core (Kato et al., 2009). When schedulers step into super medium, virtualization became a powerful tool, virtual machines can adjust the real time programs, since they allow an independent platform to develop and provide uniqueness between programs (Buyya et al., 2009).

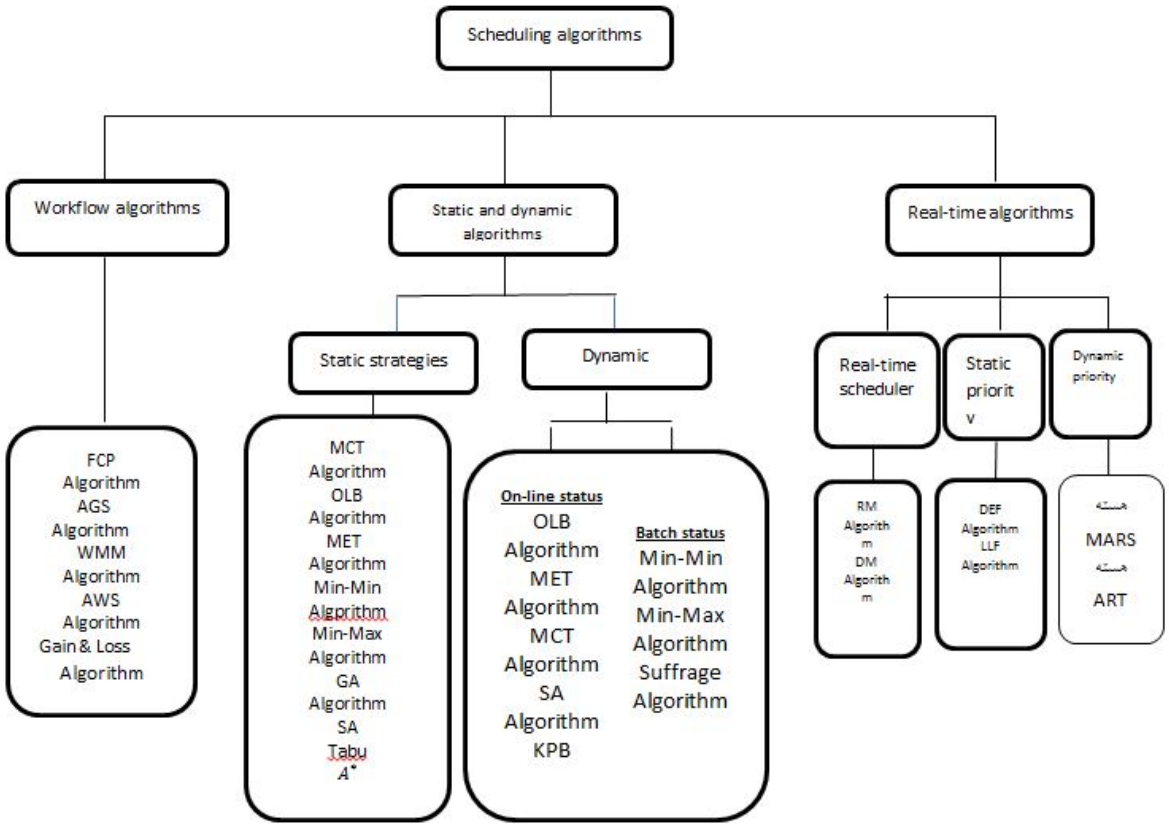

Figure 1: Types of scheduling algorithms

Conclusion

Nowadays we could possess the super-computing of advanced services only with an internet. The user can engage large computer networks consisting of thousands of small and large computers, only with an ATM card or even by means of leasing other sources. The work schedule and load distribution is one of the key issues in administrative management, which can divide various tasks among computers, help save costs and provide the users with more sources. Super-computing is a new criterion in developing applications, and can assist workflow effectively in business process management systems. The workflow technology can manage the business process in an efficient way. In super medium, the computing sources are scheduled in a way that providers can use their sources at maximum rate and the users can access their needed applications only with a low cost. Therefore, the manner of task scheduling will be considered as an important issue, and has so much impact on super service providers. Scheduling, is the best source choice with the aim of load distribution in processors, and high productivity in sources while should minimize the response time, task completion and even service cost. There are different sorts of algorithm in this field, which some of them have been introduced in this paper.

References

[1] Aggarwal, A. and Kent, R. (2005), An adaptive generalized scheduler for grid application. High performance Computing System and Application (HPCS), 19th International Aymposium on, IEEE, pp. 188-194.

[2] Armstrong, R., Hensgen, D. and Kidd, T. (1998), The relative performance of various mapping algorithms is independent of sizable variances in run-time predictions. Heterogeneous Computing Workshop (HCW 98) proceedings IEEE, pp. 79-87.

[3] Braun, T., Siegel, H., Beak, N., Boloni, L., Maheswaran, M., Reuther, A., Robertson, J., Theys, M., Yao, B. and Hensgen, D.,(2001), A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. Joural of Parallel and Distributed computing, © Academic Press, Vol. 61, pp. 810-837.

[4] Buyya, R., Ranjan, R. and Calheiros, R. (2009), Modeling and Opportunities. International Conference on High Performance Computing & Simulation (HPCE). Leipzing: IEEE.

[5] De Falco, I., Del Balio, R., Tarantino, E. and Vaccaro, R. (1994), Improving search by incorporating evolution principals in parallel tabu search. Evolutionary Computation, IEEE world congress on computational intelligence., proceedings of the first IEEE conference on, pp. 823-828.

[6] Dong, F. and Akl, S. (2007), published. An adaptive douptive double-layer workflow scheduling approach for grid computing. High performance computing systems and applications (HPCS), 21st International symposium on IEEE, pp. 7-13.

[7] Du, Y. and Li, X., (2008), Application of workflow technology to current dispatching order system. International journal of soft computing and engineering (IJSCE), vol. 2, pp. 470-475.

[8] Freund, R., Gherrity, M., Ambrosius, S., Campbell, M., Halderman, M., Hensgen, D., Keith, E., Kidd, T., Kussow, M. and Lima, J. (1998), Scheduling resources in multi-user, heterogeneous, computing environments with smartNet. Heterogeneous computing workshop (HCW 98) Proceedings, IEEE, pp. 184-199.

[9] Hong, S., Shi-ping, C.,Chen, J. and Kai, G., (2013), Research and Simulation of Task Scheduling Algorithm in Cloud Computing. TELKOMNIKA Indonesian Journal of Electrical Engineering, Vol. 11, pp. 6664-6672.

[10] Kato, S., Rajkumar, R. and Ishikawa, Y. (2009), A loadable real-time scheduler suite for multicore platforms. Citeseer, Technical Report CMU-ECE-TR09-12.

[11] Lenng. J. and Whitehead, J., (1982), On the complexity of fixed-priority scheduling of periodic, real-time tasks. Performance evaluation, Vol. 2, pp. 237-250.

[12] Liu, C. and Layland, J., (1973), Scheduling algorithm for multiprogramming in a hard-real-time environment. Journal of the ACM (JACM), Vol. 20, pp. 46-61.

[13] Maheswaran, M., Ali, S., Siegal, H., Hensgen, D. and Freund, R. (1999) Published. Dynamic matching and scheduling of a class of independent tasks onto heterogeneous computing systems. Heterogeneous Computing Workshop (HCW'99), IEEE, pp. 30-44.
[14] Radulescu, A. and van Gemund, A. (1999) Published. On the complexity of list scheduling algorithms for distributed memory systems. Proceedings of the 13th international conference on Supercomputing, ACM, pp. 68-75.
[15] Sakellariou, R., Zhao, H., Tsiakkouri, E. and Dikaiakos, M. (2007), Scheduling workflows with budget constraints Integrated research in GRID computing Springer.
[16] Tokuda, H. and Mercer, C., (1989), Arts: A distributed real-time kernel. ACM SIGOPS Operating Systems Review, Vol. 23, pp. 29-53.
[17] Yu, J. and Buyya, R., (2005), A Taxonomy of Scientific Workflow Systems for grid Computing. ACM sigmod Record, Vol. 34, pp. 44-49.
[18] Zomaya, A. and Kazman, R. (2010), Simulated annealing techniques. Algorithms and theory of computation handbook, Chapman & Hall/CRC, 33-33.