A Hybrid Approach for Radar Beam Scheduling Using Rules and Stochastic Search by Simulated Annealing

Ji-Eun ROH\textsuperscript{a}, Member, Chang-Soo AHN\textsuperscript{b}, and Seon-Joo KIM\textsuperscript{b}, Nonmembers

SUMMARY Recently, radar resource management of multifunction radar is a challenging issue in electronically scanned array radar technology. This paper deals with radar beam scheduling, which is a core issue of radar resource management. This paper proposed stochastic scheduler algorithm using Simulated Annealing (SA) and Hybrid scheduler algorithm which automatically selects two different types of schedulers according to the radar load: Rule based scheduler using modified Butler algorithm for underload situations and SA based scheduler for overload situations. The problem of how to allocate finite available resources (e.g. power, time, etc.) in an optimal way to carry out a chosen mission. This is of fundamental importance if multifunction radar systems are to realize their full potential, and scheduler should compete to obtain resource, and scheduler should decide optimal sequences of such tasks to be executed in order to meet the performance requirements of the radar functions while observing a set of given constraints such as deadlines and available resources. There have been many previous approaches to this scheduling, using artificial intelligence techniques [3]–[6], Markov decision processes [7], [8], and quality of services [9]–[11]. Compared with simple heuristic rule based algorithm, these adaptive scheduling algorithms are much more complex, and should, theoretically, yield better performance. However, they are not mature enough for real applications because of their high computational complexity.

One important aspect of scheduling problem is radar load, because the performance of scheduling algorithm can be highly changed according to the load situation. For example, one work [1] compared two rule based algorithms [11] and [14] in terms of underload and overload situation, respectively. [15] also showed that a few wrong scheduling results in the overload situation can degrade radar total performance like domino effects.

Our study is motivated from the intuition that if radar system is in underload situation, adaptive optimized scheduling algorithm is not necessary, because the problem is simple and the effect of optimization does not make a big difference in terms of overall radar performance. However, if the radar system is in overload situation when there are not enough resources to perform all requested tasks, radar strongly requires smart scheduler to control radar load and thus minimize its performance degradation. For example, when a multifunction radar system tracks only one target at present, scheduler only needs to simply schedule high priority track tasks for the target on time periodically, and low priority search tasks in the rest time. In this situation there is no competition among other high priority tasks, and it is natural that highly complex adaptive scheduler cannot significantly improve the performance compared with rule based scheduler. However, if the radar meets dozens of targets, quite a number of high priority tasks for multi-target tracking should compete to obtain resource, and scheduler should decide optimal sequences of such tasks to be executed in or-
order to maximize radar performance by minimizing latency of requested tasks.

Motivated by this, this paper proposed stochastic scheduler algorithm using Simulated Annealing (SA) and Hybrid scheduler algorithm which automatically selects two different types of schedulers according to the radar load: Rule based scheduler using modified Butler algorithm for underload situation and SA based scheduler for overload situation. This paper is organized as follows. First, brief introduction to radar beam operations and terminologies are given in Sect. 2. A rule based scheduler based on modified Butler algorithm is described in Sect. 3. SA based scheduler and Hybrid scheduler are described in Sect. 4 and Sect. 5, respectively. Section 6 provides simulation results to show how different schedulers behave in different load condition. Concluding remarks about the main contribution of this work is given in Sect. 7.

2. Review of Radar Beam Operation and Terminology

Figure 1 is a notational block diagram of how a Job is generated in a multifunction radar system. Search or Surveillance jobs are generally always running. When a search results in target detection, called Plot Confirmation is re-issued to confirm target presence. Plot Confirmation (shortly, Confirmation) is ‘look back’ operation to ensure that the target is detected again with aims to distinguish search results from false alarms. Upon a successful confirmed detection, Track Initiation is undertaken to establish a track of a given accuracy in the shortest time possible. Upon a successful Track Initiation, Track Update (shortly, Track) is undertaken to track a target. A task is an instance of a job. Generally, Search job is performed by a largely number of search tasks in order to cover interest search region, while Confirmation and Track Initiation jobs derived from one search result are performed by a small number of tasks (e.g. 3–5), respectively. Track Update job for one target is performed by periodic repeated track tasks until the target is missed. In this paper, a task is regarded as scheduling unit, and the reason is explained in Sect. 3 in detail. Figure 2 shows a relationship between task and beam composed of pulses. Here, a task is performed by one radar beam, thus a task is corresponding to a beam. Generally, a beam consists of several groups of ‘look’s, and a look consists of a group of pulses.

From the perspectives of real time constrain of job in a radar system, late job execution causes degradation of radar performance. For example, if Confirmation does not execute on time, detection of new target may be deferred to the next search revisit time. If Track Initiation does not execute on time, track initiation range which is important system performance measure may be decreased. If Track Update does not execute on time, target may be missed. Additionally, if radar is operated by rotating antenna, and a certain task is scheduled lately, the task cannot be performed when a beam direction for the task is in out of boundary of antenna beam steering coverage.

3. Rule Based Scheduler

Butler algorithm described in [12] defined three level work unit of radar: job, task, and look. In scheduling, the priority of each job is fixed: (1) plot confirmation, (2) track initiation, (3) track update, and (4) surveillance; the smaller this priority number, the higher priority the corresponding job has. In Butler algorithm, a timebalance scheme controls the scheduling process of the requested tasks. Each task has a timebalance that reflects earliness or lateness of a task in terms of ‘requested time’ of tasks\(^7\). When a task is late, its timebalance is positive. On the other hand, an early task has a negative timebalance. Finally, a task due to be undertaken at a precise moment has a timebalance equal to zero. If the job table receives a new task request that should not be scheduled for \(n\) seconds, the timebalance associated to that task should be set to \(-n\) seconds. The timebalances of all tasks are incremented as time elapses. A new job is always inserted in the job table before its due time of execution and hence, with negative timebalance. The flow diagram of this scheduling algorithm is presented in Fig. 3. In this way, the algorithm schedules the ‘looks’ of a task in sequence when a look completes its execution, and scheduling of looks from a task may only be interrupted by jobs at a higher level on the priority. Even though a look is considered as a scheduling unit in original Butler algorithm, generally in terms of radar processing, a task processing result rather than look processing result is meaningful. Because, a new task is generated after completion of the last task processing rather than a look.

\(^7\)Generally, a radar task has a time parameter called ‘requested time’. Request time is time to be expected that the first pulse of the task is transmitted by radar transmitter. Simply, it is assumed that all looks of a task have the same request time.
A key concept of Butler algorithm is that a task with the highest priority and the largest positive time balance is scheduled at the earliest time. Namely, the most highly delayed task with the highest priority among waiting tasks is executed at the earliest time. In contrast, a task with the lowest priority and the least negative time balance, i.e., the earliest task with the lowest priority is executed at the latest time. Butler algorithm schedules tasks at a precise requested moment at the earliest, and the rest tasks are scheduled later than the requested time. In other words, it does not allow early scheduled task than the request time of the task. How- ever, in real radar operation, late executed task is more serious than early executed tasks. The accumulated effect of late track updates may cause track miss and late plot con- firmation may cause late track initiation, and hence lead to significant degradation in overall radar performance.

From this, we proposed modified Butler algorithm to allow tasks scheduled earlier than the requested time by inserting new steps with red dotted boxes in Fig. 4. If no task has a positive time balance then choose a task with the largest negative time balance larger than $\alpha$. Here, $\alpha$ is threshold time value to limit excess early schedule. Allowing early sched- uled tasks might be expected that enables to relieve competition among tasks which are requested at the same time. A change of scheduling unit from ‘look’ to ‘task’ is also reflected in Fig. 4. From now on, we name modified Butler algorithm rule based algorithm with the conflicting meaning of stochastic scheduler using SA in Sect. 4.

4. Stochastic Scheduler Using SA

SA was first proposed by [16] as a method for solving combinatorial optimization problems. The name of the algo- rithm derives from an analogy between the simulation of the annealing of solids and the strategy of solving combina- torial optimization problems. Annealing refers to a process of a cooling material slowly, until the material reaches a sta- ble state. Early in this process, at high temperatures, parti- cles in the material will sometimes change to higher energy states, but at low temperatures such behavior is much less likely. At very low temperatures, particles virtually always move to low energy states whenever the opportunity arises. Eventually, the movement toward low energy states leads to freezing. SA is a generalization of a Monte Carlo method for statistically finding the global optimum for multivariate functions. It has been used in operations research to success- fully solve a large number of optimization problems such as the TSP and various scheduling problems [17]–[19].

SA consists of seven phases: 1) initialize the algo- rithm parameters, such as temperature and initial solution, 2) make new solution, 3) compare two solutions by using objective function, 4) determine whether new solution is ac- cepted or not, 5) repeat phase 2 through 4 until stopping criterion is met (inner loop), 6) decrease of temperature, 7) repeat phase 2 through 6 (outer loop) until a lower temper- ature is achieved. Adopting these concept, SA based beam scheduling algorithm is proposed as shown in Algorithm 1. From Step 1 to Step 3 correspond to phase 1. Step 4, 5, 6, 7, 8, 9 correspond to phase 2, 3, 4, 5, 6, 7, respectively.

Compared with rule based scheduler selecting only one task among several waiting tasks when the scheduler is called, the main difference of SA based scheduler is to si- multaneously schedule several tasks within time interval by using SA.

To do this, several parameters should be defined. Each task has a unique ID, its requested time, priority, and dwell time needed to transmit all pulses. \textit{Scheduling Time Interval} is time interval to schedule, and Scheduling Candidate Set (SCS) are determined by current time, \textit{Scheduling Time Interval} and requested time of waiting tasks. Determination of SCS is described in Step 1 of Algorithm 1. All
tasks which belong to SCS are randomly lined up as an initial solution, called V schedule. To make a new solution, called V’ schedule, select random two tasks and exchange their order each other (Step 4). These two schedules are evaluated by cost function $Cost_Calc$ (Step 5) and determined whether V’ is accepted or not instead of V (Step 6). These steps are repeated until stopping criterion is met.

Figure 5 shows the conceptual diagram to explain SA based scheduling algorithm.

A key part of SA based scheduler is definition of algorithm parameters and cost function. $Scheduling\ Time\ Interval$ is one of the important parameters, which determines scheduling interval for optimization. If scheduling interval is long, quite a number of tasks to be determined their sequence feed into input of SA, namely SCS. In this case, SA needs much time to converge, and it is difficult to respond rapidly to unexpected urgency tasks, such as confirmation. On the other hand, if the scheduling interval is too short, only a few tasks feed into input of SA, and hence stochastic search to obtain an optimal sequence might be meaningless.

For cost function to evaluate scheduling optimality, earliness and lateness are adopted. Earliness penalty and lateness penalty are calculated for each task and summed on V schedule. To sum up, SA based scheduler forms SCS in every scheduling time interval, and an optimal schedule V.

Interval is one of the important parameters, which determines scheduling interval for optimization. If scheduling interval is long, quite a number of tasks to be determined their sequence feed into input of SA, namely SCS. In this case, SA needs much time to converge, and it is difficult to respond rapidly to unexpected urgency tasks, such as confirmation. On the other hand, if the scheduling interval is too short, only a few tasks feed into input of SA, and hence stochastic search to obtain an optimal sequence might be meaningless.
for SCS is solved in the direction of minimizing earliness and lateness penalty through annealing process.

5. Hybrid Scheduler Reacting to Radar Load Situation

Here, we describe radar beam schedulers called Hybrid scheduler which automatically selects two different types of schedulers according to radar load: Rule based scheduler using modified Butler algorithm for underload situation and SA based scheduler for overload situation. To do this, radar load should be calculated within every scheduling interval. The detailed description of the algorithm is given Algorithm 2.

Hybrid scheduler calls SA based scheduler or rule based scheduler according to the radar load calculated in every scheduling interval. If the load exceeds threshold value \( \beta \), SA based scheduler is executed, otherwise rule based scheduler is executed. Radar load is calculated by summing dwell times of tasks requested within interval and dividing it by interval length. Noted that radar load addressed here is calculated from dwell times of tasks except for search tasks, and hence it corresponds to radar load except for surveillance.

Generally, radar system has a requirement about minimum required occupancy for surveillance to guarantee search performance. For example, surveillance radar devotes most of the time to surveillance, whereas tracking radar devotes most of the time to tracking. Concerning this, in our algorithm, \( \beta \) is set to 0.5. This means if radar time is devoted more than 50% of total time to tasks except for search tasks, overload situation is declared.

Concerning \( Scheduling\ Time\ Interval \), a relationship between \( Scheduling\ Time\ Interval \) of SA based scheduler and that of Hybrid scheduler is like that:

\[
Scheduling\ Time\ Interval_{Hybrid} = N \times Scheduling\ Time\ Interval_{SA}
\]

where \( N \) is an integer.

This means that SA based scheduling is performed \( N \) times within \( Scheduling\ Time\ Interval_{Hybrid} \) in Hybrid algorithm.

6. Simulation

6.1 Simulation Setup

Simulation setup is the same as CH5 in [12] in order to compare the simulation result of Butler algorithm. The radar mission is to cover two surveillance regions with the identical azimuth span \( \pm 45 \) but varying elevation spans, \( 0^\circ-20^\circ \) (region 1) and \( 20^\circ-50^\circ \) (region 2) in elevation.

Table 1 shows parameters of the two search regions. In this simulation we assumed if target is detected during searching two regions, then the target is confirmed by plot confirmation tasks, is initiated by track initialization, and is tracked by periodic track update tasks, sequentially. In other words, a new task except for a search task is generated according to the result of completion of previous scheduled task, as shown in Fig. 1.

In this simulation, 70 targets randomly appear until 30s after simulation start, and simulation is continued until 50s. The tracking period is set to be 0.2s; the intervals of plot confirmation and 10 successive tracking initiations are 0.1s. The target detection probability is 0.9 and the false alarm probability is \( 10^{-6} \). A more realistic surveillance volume is constructed by allowing the surveillance dwell times to increase with scan angle, to compensate for the loss in gain with coherent integration of pulses. Signal to noise ratio

<table>
<thead>
<tr>
<th>Region</th>
<th>Azimuth</th>
<th>Elevation</th>
<th>Avg. dwell time (ms)</th>
<th>Occupancy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( \pm 45^\circ )</td>
<td>( 0^\circ-20^\circ )</td>
<td>1.42 broadside:1</td>
<td>20%</td>
</tr>
<tr>
<td>2</td>
<td>( \pm 45^\circ )</td>
<td>( 20^\circ-50^\circ )</td>
<td>3.304 broadside:2</td>
<td>80%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Region</th>
<th># of beams</th>
<th>Frame Time(s)</th>
<th>Function time(s)</th>
<th>Look interval(ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>283</td>
<td>2.010s</td>
<td>0.402</td>
<td>7.102</td>
</tr>
<tr>
<td>2</td>
<td>329</td>
<td>1.359s</td>
<td>1.087</td>
<td>4.130</td>
</tr>
</tbody>
</table>
SNR\(_\varphi\) of received signal is defined below [13]:

\[
\text{SNR}_\varphi = \text{SNR}_0 \cos^3(\varphi)
\]

where the scan angle \(\varphi\) is the angle between beam direction and the normal vector of array plane and SNR\(_0\) is signal to noise ratio on the array broadside. The dwell time of each beam is increased by inverse of reduction in signal to noise ratio with coherent integration of pulses so that the detection performance can be maintained. Additionally, function time \(t_{\text{function}}\), frame time \(t_{\text{frame}}\), and look interval \(t_{\text{look}}\) shown in Table 1 are defined like this:

- \(t_{\text{function}} = t_d \times N_{\text{beam}}\)
- \(t_{\text{frame}} = t_{\text{function}} / \text{occd}\)
- \(t_{\text{look}} = t_{\text{frame}} / N_{\text{beam}}\)

where occ\(_d\), \(N_{\text{beam}}\), and \(t_d\) are desired occupancy, the number of beams, and the average dwell time for the surveillance region, respectively. From the viewpoint of simulation model validity, as mentioned earlier, this simulation model was designed in [12] for both validation of scheduling algorithm of MESAR experimental multifunction radar and ease analysis of results. This simulation model has been made intentionally very similar to the architecture used in real radar system MESAR\(^1\). Additionally, one work [1], which compared two radar scheduling algorithms, also adopted the simplified version of the same simulation model.

Parameters for rule based scheduler, SA based scheduler, and Hybrid scheduler are given below:

- \(\alpha = 1\)ms\(^{11}\)
- \(\text{Scheduling Time Interval} = 10\)ms
- \(\text{max}\_\text{annealing time} = 7\)ms
- Initial temperature parameter, \(T = 10\);
- Temperature length \(L = 5\);
- Cooling rate \(F = 0.95\);
- Boltzmann factor \(k = 1\);
- \(\text{Penalty}_\text{earliness} = [10 20 30 0]\)
- \(\text{Penalty}_\text{lateness} = [30 10 15 0]\)
- \(\beta = 0.5\)

\(\text{Scheduling Time Interval}\) is common to SA based scheduler and Hybrid Scheduler. Given \(\text{Scheduling Time Interval}\) is 10ms, average 11 tasks feed into input of annealing process at a time. Noted that all ordered tasks of SCS determined from SA are not executed by radar within 10ms. In detail, ordered tasks from SA are sequentially executed until sum of their dwell time is nearly 10ms. In our simulation, average 6 tasks among average 11 tasks are executed in every time interval. The rest tasks are pending and have another chance in next time interval. This scheduling interval was determined after performing pre-experiment to investigate how different time intervals have an effect on scheduling performance. With scheduling interval, maximum annealing time, which is one of termination condition of annealing process, is also important parameter in terms of affordability and degree of completeness of SA based scheduler. Here, \(\text{max}\_\text{annealing time}\) is set to 7ms in order to ensure 30% of margin over scheduling interval until next scheduling time.

Concerning penalty, earliness penalty of track task is highly imposed in order to prohibit excessive resource allocation according to frequent track update resulted from early scheduling. On the other hand, lateness penalty for plot confirmation task is highly imposed. Generally, after a target has been detected in a search dwell and the target is not yet in track, a dwell is transmitted in the direction measured by the search dwell to confirm the presence of a target. A successful confirmation results in the initiation of a track. The delay in the transmission of a confirmation dwell must be short to ensure that the target is still within half-a-beamwidth of the direction measure by the search dwell. This is the reason why the lateness penalty for plot confirmation is highly imposed. However, these penalties can be changed in any degree according to radar mission or radar design policy. Concerning load threshold \(\beta\) set to 0.5, overload situation is declared to scheduler when the radar time is devoted more than 50% of total time to tasks except for search tasks. In our simulation, scheduling on the look level is not considered, because dwell time is too short.

### 6.2 Simulation Results

Three aspects are assessed to evaluate the performance of three schedulers. These are (i) latency reflecting degree of earliness and lateness between requested time and executed time, (ii) the number of executed tasks during scenario time, and (iii) scheduling time required to perform scheduling.

(1) Latency of each scheduling algorithm

The main goal of scheduler is to schedule radar tasks close to their requested (desired) time of execution. From this viewpoint, latency, which is common measure to evaluate scheduling performance, reflects degree of earliness and lateness between requested time and executed time of task. If a task is executed early than requested time, latency becomes negative value with amount of the time differences; otherwise if a task is executed late, latency becomes positive value with amount of the time differences. Figure 6 depicts latency histogram when using original Butler algorithm and modified Butler algorithm. As mentioned earlier, Butler algorithm schedules tasks with positive timebalance value first, and hence some tasks are scheduled on time, and the rest tasks are scheduled later than the requested time. Figure 6 shows that 150 tasks are scheduled on time in orig-
inal Butler algorithm, whereas 323 tasks are scheduled on time in modified algorithm. The results suggest that modified algorithm can schedule many more tasks on time by allowing early scheduled task than the requested time. This implies that allowing early scheduled tasks succeed in relieving competition among tasks requested at the same time. In modified Butler algorithm, 1.6% tasks are scheduled on time and 6.8% tasks are scheduled within ±1ms latency.

Figure 7 shows latency histogram according to load status in modified Butler algorithm. In underload status, 12% tasks are scheduled on time, and 46% of total tasks are scheduled within ±1ms latency. The highest distribution of histogram is also placed on 0ms. On the other hand, in overload status only 1% tasks are scheduled on time and 6% tasks are scheduled within ±1ms latency, while 41% tasks are scheduled within a range of 4ms and 6ms latency. This shows a performance of rule based scheduler using modified Butler algorithm is degraded as radar load increases.

Figure 8 shows latency histogram when using SA based scheduler. Compared with results of rule based scheduler, 6.6% tasks are scheduled on time and 23% tasks are scheduled within ±1ms latency in SA based scheduler. Thus, it is obvious that SA based scheduler has a very good ability to schedule many more tasks on time. Additionally, the number of scheduled tasks in SA based scheduler is much higher than the number of tasks in rule based scheduler. An analysis of number of scheduled tasks is discussed in later in detail. Figure 9 shows latency histogram according to load status in SA based scheduler. From this figure, it is observed that 36% tasks are scheduled within ±1ms latency in underload status, and similarly 36% tasks are scheduled within ±1ms latency in overload status. The distribution shapes in the overload and underload status are also resembled. Compared with results of rule based scheduler shown in Fig. 7, robust performance regardless of load status is the most distinctive characteristic of SA based scheduler.

This suggests that gradual performance degradation is achieved in overload situation when using SA based scheduler.
Figure 10 presents latency histogram when using Hybrid scheduler. From this figure, it is observed that 7% tasks are scheduled on time and 24% tasks are scheduled within ±1ms latency. Figure 11 shows latency histogram according to load status in Hybrid scheduler. In underload status 12% tasks are scheduled on time and 46% tasks are scheduled within ±1ms latency. In overload status 10% tasks are scheduled on time and 37% tasks are scheduled within ±1ms latency. Compared with the result of SA based scheduler, the performance of Hybrid scheduler is slightly better, but the difference is not significant.

Table 2 summarizes latency performance described so far. From the results presented above, it is observed that rule based scheduler is faced with difficulty in dealing with overload situation.

However, a desirable behavior was observed in the case of SA based scheduler and Hybrid scheduler. Results of the simulations suggest that both two schedulers have an ability to schedule many more tasks on time even in overload situation.

Table 2  Comparison of latency performance for each scheduler.

<table>
<thead>
<tr>
<th></th>
<th>On time</th>
<th>within ±1ms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule based</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Underload</td>
<td>12%</td>
<td>46%</td>
</tr>
<tr>
<td>Overload</td>
<td>1%</td>
<td>6%</td>
</tr>
<tr>
<td>Total</td>
<td>6.6%</td>
<td>23%</td>
</tr>
<tr>
<td>SA based</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Underload</td>
<td>8%</td>
<td>36%</td>
</tr>
<tr>
<td>Overload</td>
<td>10%</td>
<td>36%</td>
</tr>
<tr>
<td>Total</td>
<td>7%</td>
<td>24%</td>
</tr>
<tr>
<td>Hybrid scheduler</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Underload</td>
<td>12%</td>
<td>46%</td>
</tr>
<tr>
<td>Overload</td>
<td>10%</td>
<td>37%</td>
</tr>
</tbody>
</table>

Figure 12 shows the change of radar load status Load during scenario period, which is returned value from Radar_Load_Monitoring function in Hybrid scheduler algorithm. This indicates radar time occupancy devoted to all tasks except for search task within scheduling interval. From this, we can understand scenario complexity from the viewpoint of radar time and we can know when rule based scheduler and SA based scheduler are called, respectively. After scenario starts, total radar time is devoted to search tasks initially, decreases to 50% before about 13s, and finally decreases to fewer than 15%. Thus, Hybrid scheduler schedules tasks using rule based scheduler until around 13s and schedules task using SA based scheduler at the rest time. Load increases continuously until 35s and then it is maintained, because target generation is finished around 30s and 70 tracks are updated periodically without new target detection in the rest time.

(2) The number of scheduled tasks in each scheduling algorithm

In [15], the goal of radar beam scheduling is to maximize the number of scheduled tasks and insure that non scheduled or delayed tasks have lowest priority. From the view
point of this, another aspect analyzed here is the ability about how many tasks are scheduled during scenario periods. Table 3 shows the number of scheduled tasks according to applying each algorithm for the same scenario simulation. This result is obtained by averaging 20 simulations. During scenario time 50s, it is seen that SA based scheduler and Hybrid scheduler can schedule many more tasks than rule based scheduler. Even though the number of scheduled tasks by Hybrid scheduler is slightly less than that by SA based scheduler, the difference is not significant. The number of scheduled tasks indicates the amount of tasks radar performs. From the examinations presented, it may be inferred that the radar system have a chance to detect more targets or to track targets more frequently by performing many more tasks when using Hybrid scheduler and SA based scheduler than rule based scheduler.

(3) Time complexity of each algorithm

The third aspect of the comparison among the scheduling algorithms is ability about how much time is required for scheduling. As noted earlier, compared with simple heuristic rule based algorithm, adaptive scheduling algorithms based on optimization are many more complex, even though theoretically it yields better performance. Thus, they are not mature enough for real applications so far because of their high computational complexity.

Table 4 summarizes maximum spent time to be needed for each scheduling algorithm performs. This CPU time is measured in the PC environment of Intel(R) Core(TM)2 Quad CPU Q8300 @2.50 GHz, 3.50 GB RAM, and simulation language is Matlab R2009b.

Rule based scheduler scheduling one task at a time requires 0.1ms to dispatch rules. Time complexity of SA based scheduler is highly related with termination condition of annealing and the length of scheduling interval. As mentioned earlier, max_annealing_time was set to 7ms in order to guarantee 30% of margin time until next scheduling time. It is seen that SA based scheduler requires max 7ms and avg. 6ms to schedule avg. 11 tasks at a time in every scheduling interval. Considering that scheduling interval was set to 10ms, as expected it can have a margin time about minimum 3ms until next scheduling time. In Hybrid scheduler, rule based scheduler is called 5114 times, and SA based scheduler is called 2514 times. Additional overhead such as execution of Radar_Load_Monitoring function requires avg. 0.2ms and it is not significant. Concerning total scheduling time, Hybrid scheduler can save 24% time for scheduling compared with SA based scheduler. Considering our simulation program code is not optimized, it is expected that real computational time in COTS board with code optimization is much less than this measured time. This analysis suggests that computational load is of SA based scheduler is affordable enough to execute in a real radar system.

To sum up, rule based scheduler is simple and robust in underload situation, whereas its performance is significantly degraded as load increases. However, SA based scheduler guarantees robust performance regardless of load situation. It can schedule many more tasks on time for the same operation period even in the overload situation. Even though its time complexity is relatively high, it is affordable to be applied for real application by controlling parameters. From the view point of load with which radar is faced, in underload situation where there are not many high priority tasks that had conflicting transmitting times, rule based scheduler is sufficient and SA based scheduler is not necessary, rather it increases time complexity. However in overload situation SA based scheduler is strongly needed to guarantee gradual performance degradation. Hybrid scheduler, motivated from this, can provide robust solution reacting to radar load with advantage of relatively short scheduling time.

7. Conclusion

This paper proposed modified Butler algorithm, SA based scheduler algorithm, and Hybrid scheduler algorithm using two proposed algorithms according to radar load. Performance of the algorithms was analyzed in terms of latency, the number of scheduled tasks, and time complexity. Compared with original Butler algorithm, simulation results show that modified algorithm can schedule many more tasks on time by allowing early scheduled task. While the performance of rule based scheduler is degraded in overload situation, SA based scheduler adopting earliness and lateness penalty guarantees robust performance in overload situation. Hybrid scheduler also provides comparable performance with the advantage of low time complexity. As a future work, the proposed algorithms will be evaluated in terms of the overall radar performance perspectives, such as track capacity and track initiation range with search performance.

References


Ji-Eun Roh received her B.S. degree in Computer Science and Engineering from Pusan National University (PNU), and then her M.S. and Ph.D. degree in Computer Science and Engineering from Pohang University of Science and Technology (POSTECH), in 2002 and 2006, respectively. From 2006, she has been with Agency for Defense Development (ADD) as a senior researcher.

Chang-Soo Ahn received the B.S. and M.S. degrees in Electrical Engineering from Korea University in 2002 and 2005, respectively. From 2005, he has been with Agency for Defense Development in Rep. of Korea.

Seon-Joo Kim received the B.S. and M.S. degrees in Electrical Engineering from Aju University in 1986 and 1988, respectively. From 1988, he is with Agency for Defense Development in Rep. of Korea.