Using a Task-Specific QoS for Controlling Sensing Requests and Scheduling

Yong Xun
Mieczyslaw M. Kokar
Department of Electrical and Computer Engineering
Northeastern University
Boston, MA 02115
yxun@ece.neu.edu; kokar@coe.neu.edu

Kenneth Baclawski
College of Computer and Information Science
Northeastern University
Boston, MA 02115
kenb@ccs.neu.edu

Abstract

Typically, management of networked computational and sensing nodes is based upon a quality of service metric (QoS) that is based on some generic principles, like "be fair in allocating resources" or "utilize the CPU capacity to the maximum". The consequences of accepting such a starting point is that (1) task-specific resource requirements are not taken into consideration, and (2) computational and communication resources are saturated without paying attention to whether such a high load is necessary or not. In this paper we describe some of our efforts on how to improve the situation described above. In particular, we discuss one of the approaches that we are currently investigating that can be summarized by the following three points. (1) We use a task-specific QoS (TS-QoS) as a variable that is controlled by our system. (2) Requests for resources are generated based upon the feedback provided by the TS-QoS, where the request generator's parameters are adjusted using a simple PID controller. (3) A Dynamic Programming based algorithm is used for scheduling resource. Simulations using sensor resources show some of the advantages of the proposed approach.

1. Introduction

The goal of management of networked computational and sensing nodes is to attempt to optimize performance as measured by a quality of service metric (QoS) (cf. [6, 1, 2, 3, 5, 8]). Performance is most commonly measured using generic principles which need not be relevant in a particular case. For performance measurements to be meaningful, the QoS should be specific to the particular domain and task being performed. A QoS with this property is called a *task-specific QoS* (TS-QoS).

To investigate TS-QoS based resource management we selected a scenario in which there is one resource (a sensor) whose goal is to monitor n abstract objects (radar illuminations). The sensor needs to be directed to tune to a specific frequency (of the radar) at a specific time. Since the object (the illumination) is visible only at some times, i.e., whenever the radar is illuminating in the direction of the sensor, a generic QoS metric, like the amount of time that the sensor is allocated to a particular radar, will not work. It is the timing of the tuning that matters the most. The sensor is located on a moving platform (aircraft). The workload of the sensor is defined by the number of emitters in the environment and their illumination times. The TS-QoS represents the uncertainty about the states of the emitters.

In general, we deal with algorithms of soft real-time, non-preemptive resource scheduling for a scenario with the following characteristics: (1) The environment is non-deterministic, i.e., the scheduler does not know the state of the environment and the value of the gain from scheduling a particular task at a given time. (2) The problem is dynamic in the sense that the value of the gain depends not only on the scheduling decision but also on the state of the environment and the passage of time. (3) The system is overloaded, i.e., the demand (the *workload*) for the resource exceeds its physical capacity.

The consequence of these constraints is that the sched-

uler needs to estimate future states of the environment, which in a sense is equivalent to estimating future tasks. Moreover, since the system is overloaded, the scheduler needs to optimize its scheduling decisions. And since the environment is dynamic, the schedule needs to adapt to changing demands.

The resource scheduling problem described above is a case of the problem called *sensor management (SM)* (cf. [7]). The specific feature of the SM problem is that the scheduler must not only schedule tasks, but also generate tasks (sensing requests). The TS-QoS then depends on both the task generation step and the scheduling step.

In this study we considered two kinds of task generation approach: periodic task generation and control-based task generation. In the periodic task generation approach, to ensure the detection of each illumination (without any prior knowledge of the occurrence of the illumination) the receiver must be tuned to a given radar's frequency every T_{eit} time units, where T_{eit} is the duration of the illumination (in a given direction) for a given radar. The stream of generated tasks according to this policy shows a fixed pattern over time. It does not depend on any feedback from the detector, i.e., the pattern remains fixed independently of whether a given radar was detected or not. In the control-based task generation approach (cf. [11]), the pattern is updated after receiving feedback from the detector.

The main goal of this paper is to propose and analyze various approaches to scheduling appropriate to the characteristics of the problem listed above. It is known [4] that the periodic task generation approach (combined with Earliest Deadline First (EDF) scheduling) is computationally feasible only for low workloads [10]. Our goal is to find other approaches that push the solvability of this scheduling problem to higher workloads while maintaining the same level of the TS-QoS. In this paper we compare four solutions with two different kinds of task generation and scheduling: (1) periodic task generation, EDF scheduling; (2) periodic task generation, DP scheduling; (3) control based task generation, TS-QoS-based greedy scheduling (4) control based task generation, DP scheduling.

This paper is organized as follows. First, in Section 2 we briefly describe our simulated scenario. This is followed by the description of the algorithms used to estimate the state of the system (Section 3). In Section 4 we briefly describe our QoS measure and the benefit function used for optimization. In Section 5, we define our problem as a constraint satisfaction problem. This is followed by the presentation of the methods of task generation, i.e., periodic task generation and control based task generation, described in Sections 6 and 7. An analysis of the time complexity of the algorithms used in our study is given in Section 8. Finally, we present results of our simulations in Section 9 and conclusions and future work in Section 10.

2. Scenario

In our simulations we consider a scenario with one sensor located on an aircraft moving in a straight line with constant velocity. The number of radars in the environment ranges from 10 for a light workload up to 120 for a heavy workload. The radars differ in various parameters, like emission frequencies, illumination times and illumination periods. As the aircraft moves, some radars are within the detection range and some others are outside the detection range. It is assumed that the the frequencies of the radars, the lengths of their illuminations, the illumination periods and the required dwell times are known, but illumination times, radar locations and phases are not. In other words, the receiver knows what kinds of radar to expect, but it does not know when and where. In order to detect a radar, the receiver needs to tune to the radar's frequency (*dwell* on it) for a long enough time. The dwell must overlap with the radar's illumination in the aircraft's direction.

3. State Estimation

Since the success of a schedule, i.e., detection of a radar illumination, depends on the knowledge of the direction (phase) of the radar illumination $\phi(i,t) \in [0, 2\pi]$, we are interested in estimating this variable. Towards this aim we model each of the emitters i at time t as follows.

$$\frac{d}{dt}\phi(i,t) = 2\pi/T_{eip} + w_\phi(i) \tag{1}$$

In this equation T_{eip} is the illumination period of the emitter and $w_\phi(i)$ is the noise. Noise is assumed to be white Gaussian with zero mean. In the experiments, we set the initial value of ϕ to 0 and then compute the values of $\phi(i,t)$ from this model.

We then discretize the state variable ϕ so that our state space is finite. Θ_m represents the estimate of the current state.

$$\Theta_m = \lceil \frac{\phi(i,t_s)}{2\pi} \cdot 360 \rceil \tag{2}$$

The resulting discrete space then is:

$$S = \{\Theta_1, ..., \Theta_K\} \tag{3}$$

where K is equal to 360. Since the value of Θ is not known, we treat Θ as a random variable with probability distribution

$$P(\Theta = \Theta_1), \ldots, P(\Theta = \Theta_K) \tag{4}$$

where $P(\Theta = \Theta_j)$ is the probability of illumination in the direction Θ_j. We initialize this distribution to be uniform, i.e.,

$$P(\Theta = \Theta_j) = \frac{1}{K},\ j = 1, 2, ..., K \tag{5}$$

Note that other ways of modeling the lack of initial knowledge can be used.

The estimation process consists of two steps: computing the *aging effect*, i.e., the effect of time on the certainty of the information about the phases of particular radars, and updating the probability distribution after an observation event.

SubSectionThe Aging Effect

When the receiver is not tuned to a particular emitter for some time, the certainty of the phase of that emitter should decrease. Essentially, we expect that in the limit, if the system does not receive any detection events (updates) the distribution of Θ should become uniform. We call this the *aging effect*. To account for this effect we update the distribution every illumination time of a given emitter according to the following algorithm.

First, we perform convolution ($\otimes$) of the distribution of Θ with white noise $N_w(X)$:

$$\widehat{P}(\Theta) = P(\Theta) \otimes N_W(X) \tag{6}$$

where the index W in N_W represents the window size. In our experiments we set $W = B_e$, the beam width of the emitter. The value of $\widehat{P}(\Theta)$ is then used to compute the new distribution according to the following formula:

$$P(\Theta = \Theta_j) = \widehat{P}(\Theta = \Theta_{j+\lceil W/2 \rceil}) \cdot \frac{1}{1-\delta} \tag{7}$$

where

$$\delta = \sum_{i=1}^{\lfloor W/2 \rfloor} \widehat{P}(\Theta = \Theta_i) + \sum_{i=K+\lceil W/2 \rceil}^{K+W-1} \widehat{P}(\Theta = \Theta_i) \tag{8}$$

3.1. Bayesian Update

Whenever the receiver is tuned to a given emitter's frequency, one of two events (E_m) can occur: *detection* at Θ_m (we denote it as D_m) or *non-detection* (we denote it as L_m). Θ_m is computed according to Eq. 2 with ϕ obtained from the dynamic model (Eq. 1). The probabilities of illumination are then updated for this emitter. We use Bayesian updating (cf. [9]):

$$P(\Theta = \Theta_j | E_m) = \frac{P(E_m | \Theta = \Theta_j) \cdot P(\Theta = \Theta_j)}{\sum_{k=1}^{K} P(E_m | \Theta = \Theta_k) \cdot P(\Theta = \Theta_k)} \tag{9}$$

The *posterior* probability depends on the *prior probability* $P(\Theta = \Theta_j)$ and the probability of a given event (detection or non-detection) $P(E_m|\Theta = \Theta_j)$. The latter probability represents the level of certainty of detection. In our case we model this dependence using two parameters: P_{fp} - the probability of false detection (or false positive) and P_{fn} - the probability of false non-detection (or false negative). For a detection event the probability is

$$P(D_m|\Theta = \Theta_j) = \begin{cases} 1 - P_{fp} & : \quad \Theta_j = \Theta_m \\ P_{fn} & : \quad \Theta_j \neq \Theta_m \end{cases} \tag{10}$$

For a non-detection event we have

$$P(L_m|\Theta = \Theta_j) = \begin{cases} P_{fn} & : \quad \Theta_j = \Theta_m \\ 1 - P_{fp} & : \quad \Theta_j \neq \Theta_m \end{cases} \tag{11}$$

If one detection/non-detection event can be associated with more than one element Θ_j of the sample space S (i.e., when the state space is discretized into increments smaller than the beam width B_e of the emitter) then the probability update formula given by Eq. 9 must be extended to incorporate this fact. For this reason, we use the following Gaussian model to distribute the posterior probability (in case of a detection event) over neighbor states of Θ_m:

$$P(\Theta = \Theta_j | D_m) = \frac{1}{\sqrt{2\pi} \cdot \sigma} \exp - \frac{(j-\mu)^2}{2\sigma^2} \tag{12}$$

where $\mu = m$, $\sigma = B_e$, and $j = 1, \ldots, K$. For a non-detection event we just use Equation 9.

4. Quality of Service

Optimizing the solution to a resource scheduling problem requires that there be some measure of quality that is being optimized. The quality of service (QoS) for sensor management formalizes the degree of uncertainty that one has about each emitter. In the absence of any observation, the uncertainty increases with time, while observations reduce the uncertainty. Observations that detect an illumination reduce the uncertainty to zero, while observations that do not detect an illumination also reduce the uncertainty but by much less. The formal specification of the QoS is the following:

$$QoS(i,t) = \frac{V(i)}{T_{eip}(i)} \cdot (t - t_d(i)) + d(i, t_l) \tag{13}$$

where

$$d(i,t_l) = \begin{cases} V(i) & t_l = T_b \\ 0 & t_l \in T_d(i) \setminus T_b \\ d(i, t_l^-) - V(i) \cdot \frac{T_{eit}(i)}{T_{eip}(i)} & t_l \in T_l(i) \setminus T_d(i) \end{cases} \tag{14}$$

In this formula, T_{eit} is the the lenght of illumination of an emitter, $t_d(i)$ is the time of last detection of emitter i, and $V(i)$ is the weight (or priority) associated with this emitter. $T_d(i)$ is the set of detection times for this emitter at the start

time T_b, and $T_l(i)$ is the set of observation times for this emitter. We consider the observation time to be the time at the end of the observation, so $T_d(i)$ is a subset of $T_l(i)$. Note that the function $d(i, t_l)$ is updated only at observation times.

To make a rational scheduling decision, we need to estimate the expected benefit from scheduling versus not scheduling of a given emitter at a particular time. We base this benefit on the value of expected gain in the QoS as a result of a scheduling decision.

If the receiver is scheduled to dwell on a given emitter i at time t_s, the expected QoS can be computed as:

$$\widetilde{QoS}(i, t_s) = P(\Theta = \Theta_m; i) \cdot QoS(i, t_s = t_d) + (1 - P(\Theta = \Theta_m; i)) \cdot QoS(i, t_s = t_l) \quad (15)$$

where Θ_m - the beam direction of the given emitter, is computed from Eq. 2 and $P(\Theta = \Theta_m; i)$ is the probability of detection of emitter i at time t_s in the direction Θ_m. This value is then the expected value of the QoS, where the first part of the equation accounts for the detection of the emitter and the second part accounts for non-detection. We can see from Eq. 13 that at detection time $QoS(i, t_s) = 0$.

Now we can define the benefit for emitter i being scheduled and being not scheduled as:

$$B(i, t_s) = \begin{cases} 0 & : \text{ not scheduled} \\ \widetilde{QoS}(i, t_s) - QoS(i, t_s) & : \text{ scheduled} \end{cases} \quad (16)$$

In the second case, the benefit is computed as the difference between the expected value of QoS when the task is scheduled and the value of QoS when it is not. In other words, if emitter i is not scheduled to be dwelled upon by the receiver at time t_s, the expected QoS can be computed directly from Eq. 13 by assuming that there are no detection or no-detection events in the given time interval $[0, t_s]$.

5. Problem Formulation

For windowing-based scheduling, tasks for a specific window are generated in advance. Scheduling decisions are made at the beginning of the window and the scheduling results are received at the end of the window. By partitioning time into a sequence of non-overlapping sub-intervals (or *time windows*), the resource management problem can be partitioned into a sequence of sub-problems. For a specific window, the tasks for the i^{th} emitter are:

$$T^i = \{T_1^i, T_2^i, ..., T_{k_i}^i\} \quad (17)$$

Each task T_j^i is defined as

$$T_j^i = ({t_r}_j^i, {t_l}_j^i, {\tau_d}^i, \omega^i) \quad (18)$$

where ${t_r}_j^i$ is the *release time*, i.e., the earliest time that this task can be scheduled, ${t_l}_j^i$ is the latest time that this task can be scheduled, ${\tau_d}^i$ is the *dwell time*, i.e., the time that the receiver needs to dwell on this emitter in order to detect it, and ω^i is the frequency band for the emitter. These parameters are defined in Section 6. The set of all tasks for a window is then:

$$T = \bigcup_{i=1}^{M} T^i \quad (19)$$

where M denotes the number of emitters.

To formulate the sensor scheduling problem we need to introduce some notation. Let ${t_s}_j^i$ denote the time of scheduling of task T_j^i. Let the benefit of scheduling be written as $B(T_j^i)$ where

$$B(T_j^i) = B(i, {t_s}_j^i) \quad (20)$$

And finally, let $\tau_d(T_j^i) = \tau_j^i$ be the dwell time for task T_j^i associated with emitter i. Using this notation, the sensor scheduling problem can be formulated as follows. Given a set of tasks T as defined in Eq. 19 for a time window $[0, W]$, choose a subset T_s of tasks from the set of tasks T, $T_s \subset T$ such that this set maximizes the benefit

$$\sum_{T_j^i \in T_s} B(T_j^i) = \max_{T_x \subset T} \sum_{T_j^i \in T_x} B(T_j^i) \quad (21)$$

subject to the satisfaction of the following constraints:

- Resource Constraint:

$$\sum_{T_j^i \in T_s} \tau_d(T_j^i) \leq W \quad (22)$$

- Non-overlapping Constraint:

$$[{t_s}_{j_1}^{i_1}, {t_s}_{j_1}^{i_1} + \tau_d(T_{j_1}^{i_1})) \cap [{t_s}_{j_2}^{i_2}, {t_s}_{j_2}^{i_2} + \tau_d(T_{j_2}^{i_2})) = \phi \quad (23)$$

where $T_{j_1}^{i_1}$ and $T_{j_2}^{i_2}$ are any two tasks in the set T_S.

As we mentioned earlier in this paper, even an optimal solution may be unsatisfactory. Consequently, we formulate our problem as a constraint satisfaction problem, rather than as an optimization problem. In such a case it would be required to find a *satisfactory* schedule, i.e., one that satisfies the condition

$$\sum_{T_j^i \in T_s} B(T_j^i, {t_s}_j^i) \geq B_0 \quad (24)$$

where B_0 is a required level of the benefit.

6. Periodic Task Generation

The periodic task generation approach is based on two assumptions. First, it is assumed that radars' illumination periods, T_{eip}, are fixed. Moreover, it is assumed that radars' beam widths, B_e, are fixed, and consequently radars' illumination times, T_{eit}, are fixed. The main idea then is to generate a task periodically with the period equal to the illumination time, T_{eit}. This idea is represented in Figure 6.

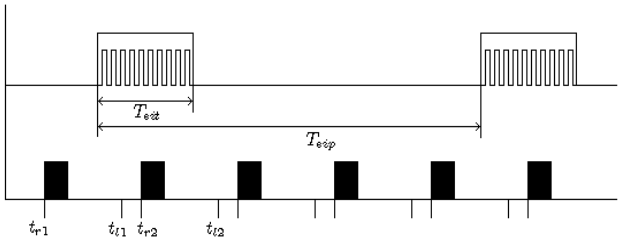

Figure 1. Periodic illumination pattern

Tasks (see Eq. 17) are generated independently for each emitter. Because of the lack of any prior knowledge, the release time for the first task, ${t_r}_1^i$ is chosen randomly. Then release times ${t_r}_j^i$ and latest execution times t_{lj}^i are computed according to the following rules:

$$ {t_r}_j^i = {t_r}_{j-1}^i + T_{eit} \tag{25} $$

$$ t_{lj}^i = {t_r}_j^i + T_{eit} - \tau_d^i \tag{26} $$

Consequently, the stream of generated tasks according to this policy shows a fixed pattern over time, although the task generation algorithm is not deterministic. It does not depend on any feedback from the detector, i.e., the pattern remains fixed independently of whether a given radar was detected or not. Note that all the tasks for the same emitter have the same ${\tau_{rv}}^i$ and ${\tau_d}^i$.

Note that, if each of the tasks generated by this approach is scheduled, i.e., if the receiver tunes to a given radar's frequency once every illumination time, the receiver is assured detection of each illumination (modulo probability of non-detection).

7. Control Based Task Generation

While the periodic task generation approach guarantees very good results, i.e., detection of each radar illumination when all the tasks generated by this approach are scheduled, it saturates the system very quickly. Note that from all the dwells only the ones that coincide with the radar illumination can actually detect the illumination. In other words, the best we can do is to have one dwell every illumination period, T_{eip}, rather than every illumination time, T_{eit}. For instance, assuming that the beam width of the radar is 2^o, only one of the 180 dwells is able to detect an illumination. If we had a perfect knowledge of the next illumination we would be able to dwell only once per illumination period and thus significantly increase the efficiency of the receiver. However, due to the various types of uncertainty present in the system, and due to the fact that this is a dynamic system (moving sensor and rotating radars) we need to have a more robust solution of this problem.

In [11] we proposed a task generator that uses feedback to control the pattern of dwells (see Figure 2). In this approach, after each dwell, two measures are computed: the QoS measure for each emitter and the instant workload. These two measures are then used as feedback to two controllers. One of the controllers adjusts the release time ${t_r}_j^i$ based upon the QoS feedback. This results in a changing pattern of task generation, and consequently in the reduction of the load on the system (fewer tasks to schedule). However this does not guarantee that the *instant workload* is within the capacity of the receiver. In other words, there still might be more tasks than the receiver can service. Accordingly, we added a second controller that controls the workload based upon the measured workload.

In this study we simply used proportional controllers (one per emitter). The output of a controller $\delta(i,t)$, was computed as

$$ \delta(i,t) = K_p(i) \cdot (P(i) - QoS(i)) \tag{27} $$

where $K(i)$ was a proportional parameter and $P(i)$ was the priority of a given emitter (we used it as the reference value of the QoS). This control output was then used by the CDW Generaror to to adjust the revisit time. Intuitively this effect can be explained as the multiplication of the controller output by the illumination period, i.e.,

$$ \tau_{rv}(i,t) = \delta(i,t) \cdot T_{eip}(i) \tag{28} $$

For the second controller we used one standard PID controller with appropriately tuned proportional, itegral and differential coefficients. Its responsibility was to compensate for changes in the demand for the workload (overload) of the receiver.

In this approach, the task pattern depends on the feedback. When the radar illumination period is constant, the pattern will eventually stabilize so that one dwell per each illumination period is generated. However, when an expected illumination is missed, the pattern becomes denser. This approach adapts the pattern to the dynamics and the uncertainty of the system.

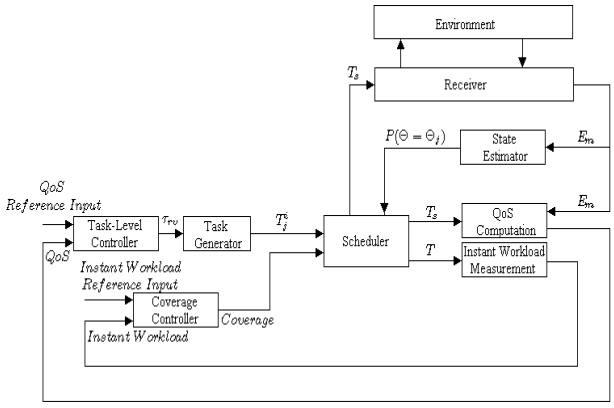

Figure 2. Control Based Task Generation

8. Complexity Analysis of Task Generation and Scheduling Algorithms

To analyze the time complexity of the algorithms we used in this study we first consider the time complexity of task generation. Since tasks are generated independently of each other, the time complexity of generating one task is $O(1)$. In the case of the periodic approach, the total complexity is $O(n)$, where

$$n = \sum_{i=1}^{M} W/T_{eit}(i) \tag{29}$$

and M is the number of emitters. The control based approach generates fewer tasks in general, but in the worst case it produces the same number, so the time complexity of control based task generation is the same as that of periodic task generation.

For scheduling, we make the simplifying assumption that every task requires the same amount of time and that the receiver capacity limits the number of tasks that can be scheduled in a window to l. Let n be the number of tasks competing to be scheduled in the window, where n is given by Eq. 29. In the case of EDF scheduling, the time complexity is $O(\min(l,n)\log n)$ because the scheduler must sort the tasks by their deadlines. Greedy scheduling based on the value of the QoS is also based on sorting so it has the same complexity. DP based scheduling, on the other hand, is more complex because it must evaluate the metric being optimized on a large number of permutations of the tasks. Assuming that $n > l$, one must evaluate every permutation of every l-element subset of the task set. The number of l-element subsets of the task set is

$$\frac{n!}{(n-l)!l!}$$

and there are $l!$ permutations of each of these subsets. Accordingly, the complexity of DP based scheduling in this case is:

$$O(\frac{n!}{(n-l)!}).$$

If $n \leq l$, then all tasks can be scheduled, so one only needs to determine the order in which the tasks should be executed. Therefore the complexity is $n!$. The complexity is even higher if we allow for the possibility of scheduling fewer than l tasks during the window. For example, if there can be a single gap in the schedule, then the gap is combinatorially equivalent to another task, so that the complexity in this case is

$$O(\frac{(n+1)!}{(n+1-l)!})$$

when $n+1 > l$ and $(n+1)!$ when $n+1 \leq l$.

9. Simulation Results

To study algorithms for resource scheduling we simulated the scenario described in Section 2. In all simulations we assumed that the required level of QoS is:

$$QoS \leq 1 \tag{30}$$

The comparison of the four algorithms used in our study is given in Figures 3 and 4. The four curves represent the following combinations of task generation and scheduling: (1) fixed pattern with EDF scheduling (top curve), (2) fixed pattern with DP scheduling (second from top), (3) control-based pattern generation with QoS based greedy scheduling (third from top), (4) control-based pattern generation with DP scheduling (bottom).

In the first case the load was light (0.9837). As we can see from Figure 3, all of the four approaches give satisfactory results, i.e., the QoS is under the required bound of 1. However, the two approaches based upon periodic task generation are better than those based on the control-based task generation. This is an expected result, since in the former case tasks are generated very densely, i.e., at each illumination time of each radar. Consequently, since the receiver can serve each of the tasks, it can detect each of the illuminations. Moreover, since according to the definition of the QoS (see Eq. 13) it decreases by certain amount after each dwell, regardless of whether an illumination was detected or not, the high density of tasks generated by the periodic approach pushes the value of mean QoS down. This is not the case for control based task generation, since in that case the tasks are generated more sparsely and thus, even though the detection rate is very good, the effect described above does not take place.

In the simulations shown in Figure 4 the load was heavy (3.4495). As we can see from this figure, for this load

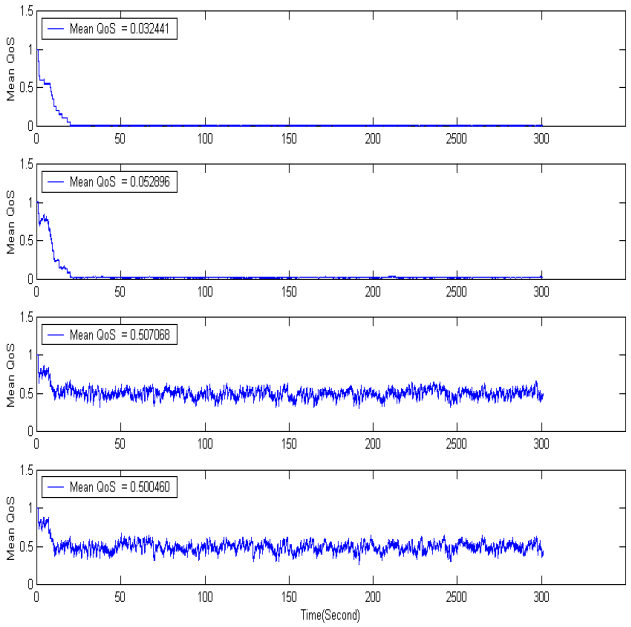

Figure 3. Comparison of performance of algorithms for light workload (0.9837)

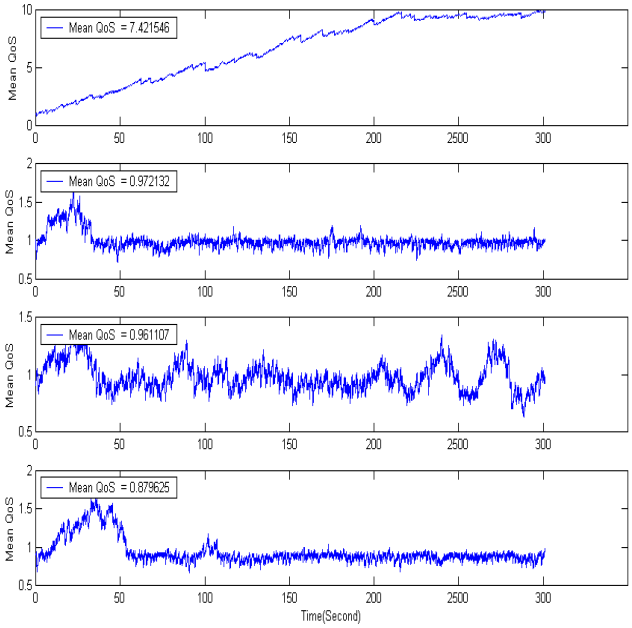

Figure 4. Comparison of performance of algorithms for heavy workload (3.4495)

the fixed pattern generation with EDF scheduling system is unstable, i.e., the QoS is not bounded. The mean QoS of the control based task generation with greedy scheduling gives a bit better performance in terms of QoS than periodic task generation with DP scheduling. Note, however, that in the periodic task generation the same effect (as described above) of the lowering of the QoS due to high density of task generation takes place. Consequently, even though this approach is a bit better in terms of QoS than the periodic/DP combination, the price for this small improvement is the higher density of tasks that the receiver has to handle. The combination of control-based task generation with DP scheduling gives the best performance with the average QoS equal to 0.88, which is better than 0.96 and 0.97 for the other two approaches.

10. Conclusions

We presented four approaches to resource management for the problem described in Section 1. Each of them has its own advantages and disadvantages. The simplest one - periodic task generation and EDF task scheduling - has the advantage that it is simple and does not require much computation. However, it can handle only workloads up to 1. This approach is not applicable to scenarios with higher loads. The approach in which tasks are generated periodically and then scheduled using dynamic programming shows a better performance in terms of satisfiability of the requirements for the level of QoS, but it is more computationally involved. The third approach (control based task generation and QoS based greedy scheduling) shows some improvement in performance over the previous two. However, this approach, similarly as the first one, results in a very high density of generated tasks and thus imposes a very high load on the receiver. The best performance is achieved with the combination of control based task generation and dynamic programming based scheduling.

While the conclusion from this study is clear - use control based task generation and DP scheduling - there are various issues that can be investigated further. For instance, since this scheduling problem is known to be NP-hard, it is important to analyze the complexity of the computation of the schedule. We are studying this problem at the current time. Another aspect that should be studied is the impact of the window size on the complexity of the computation, achievable QoS and schedulable workload. All three parameters can be measured and then can be used as feedback to control the window size.

Acknowledgments

This research was partially supported by the Defense Advanced Research Projects Agency under contract F330602-99-C-0167.

References

[1] T. F. Abdelzaher and N. Bhatti. Web Server QoS Management by Adaptive Content Delivery. In *International Workshop on Quality of Service*, pages 216–225, 1999.

[2] T. F. Abdelzaher and C. Lu. Modeling and Performance Control of Internet Servers. In *39th IEEE Conference on Decision and Control*, Sydney, Australia, December 2000.

[3] S. Brandt and G. Nutt. A Dynamic Quality of Service Middleware Agent for Mediating Application Resource Usage. In *IEEE Real-Time Systems Symposium*, pages 307–317, December 1998.

[4] Bruno Dutertre. Scan Scheduling Specification and Analysis. Technical report, System Design Laboratory, SRI International, Menlo Park, CA, May 2000.

[5] D. Hull, A. Shankar, K. Nahrstedt, and J. W. S. Liu. An end-to-end QoS model and management architecture. In *IEEE Workshop on Middleware for Distributed Real-Time Systems and Services*, pages 82–89, December 1997.

[6] C. Lu, J. A. Stankovic, T. F. Abdelzaher, G. Tao, S. H. Son, and M. Marley. Performance specifications and metrics for adaptive real-time systems. In *Proceedings of the IEEE Real-Time Systems Symposium*, 2000.

[7] S. Musick and R. Malhotra. Chasing the Elusive Sensor Manager . In *Proceedings of the IEEE NAECON*, volume 1, pages 606–613, 1994.

[8] R. Rajkumar, C. Lee, J. Lehoczky, and D. Siewiorek. Practical solutions for QoS-based resource allocation problems. In *IEEE Real-Time Systems Symposium*, pages 296–306, December 1998.

[9] L. L. Scharf. *Statistical Signal Processing*. Addison-Wesley, 1991.

[10] J. A. Stankovic, M. Spuri, K. Ramamritham, and G. C. Buttazzo. *Deadline Scheduling for Real-Time Systems - EDF and Related Algorithms*. Kluwer Academic Publishers, 1998.

[11] Y. Xun, M. M. Kokar, and K. Baclawski. Control based sensor management for a multiple radar monitoring scenario. *Information Fusion: An International Journal on Multi-Sensor, Multi-Source Information Fusion*, 5,1:49–63, 2004.