한국군사과학기술학회지 제15권 제3호, pp. 322～333, 2012년 6월

학술논문 센서 · 신호처리 부문

레이더의 부하 상태에 따른 빔 스케줄링 알고리즘의 선택적 적용

Differential Choice of Radar Beam Scheduling Algorithm According to Radar Load Status

노 지 은* 김 동 환* 김 선 주*
Ji-Eun Roh Dong-Hwan Kim Seon-Joo Kim

Abstract

AESA radar is able to instantaneously and adaptively position and control the beam, and such adaptive beam pointing of AESA radar enables to remarkably improve the multi-mission capability. For this reason, Radar Resource Management(RRM) becomes new challenging issue. RRM is a technique efficiently allocating finite resources, such as energy and time to each task in an optimal and intelligent way. Especially radar beam scheduling is the most critical component for the success of RRM. In this paper, we proposed a rule-based scheduling algorithm and Simulated Annealing(SA) based scheduling algorithm, which are alternatively selected and applied to beam scheduler according radar load status in real-time. The performance of the proposed algorithm was evaluated on the multi-function radar scenario. As a result, we showed that our proposed algorithm can process a lot of beams at the right time with real time capability, compared with applying only rule-based scheduling algorithm. Additionally, we showed that the proposed algorithm can save scheduling time remarkably, compared with applying only SA-based scheduling algorithm.

Keywords : AESA Radar(능동위상배열레이더), Resource Management(자원관리), Rule-based Algorithm(규칙 기반 알고리즘), Simulated Annealing(시뮬레이티드 어닐링), Radar Load(레이더로드)

1. 서 론

3차원 다기능레이더(MFR : Multi-Function Radar)는 실시간으로 다수의 표적을 탐지, 추적하여 거리 및 각도 정보를 제공하는 최신 레이더 시스템이다. 표적의 탐지로부터 추적까지를 실시간으로 처리하기 위해서는 빔 조향이 용이한 위상배열 안테나 기술과 다양한 실시간 레이더 신호처리 기술과 더불어, 레이더의 한정된 자원을 효과적으로 활용하여 레이더 전체 성능을 향상시키는 레이더 자원관리(RRM : Radar Resource Management) 기술의 중요성이 크게 대두되었다. 특히, 능동위상배열레이더(AESA radar : Active Electronically

† 2012년 1월 27일 접수～2012년 5월 4일 게재승인
* 국방과학연구소(ADD)
책임저자 : 노지은(jeroh@add.re.kr)

Scanned Array radar)는 전자적으로 빔을 조향함으로써 빔조향 시간이 비약적으로 빨라, 기존의 기계식 빔 조향 레이더에 비해 레이더에서 수행할 수 있는 다중 임무 처리 능력이 크게 향상되었다. 이러한 다중 임무 처리 능력을 극대화하기 위해서는 레이더에 주어진 시간, 에너지, 처리능력 등의 한정된 자원을 실시간으로 효율적으로 관리, 운용할 수 있는 레이더 자원관리 기술이 필요하다.

레이더의 한정된 자원을 효율적으로 사용하기 위해 적응적 파형 선택, 표적 위협도 분석을 통한 표적별 차등 처리, 적응적 추적 주기 기법 등이 연구 되고 있으며, 그 중 레이더 빔 스케줄링 기술은 레이더 자원관리의 핵심적인 요소라 할 수 있다. 레이더 빔 스케줄링은 레이더 임무들의 수행 시간과 순서를 임무의 우선 순위, 전술적 요구조건 및 임무 처리 성능 지표에 입각하여 실시간으로 결정하는 기술이다.

본 논문에서는 규칙 기반의 빔 스케줄링 알고리즘과 SA(Simulated Annealing)을 이용한 빔 스케줄링 알고리즘을 소개하고, 이를 시시각각 변하는 레이더 부하 상태를 고려해 빔 스케줄링 알고리즘을 선택하는 방법을 제안, 그 효용성을 분석하였다. 본 논문의 II장에서는 다기능레이더 시스템을 위한 레이더 자원관리 연구 중, 빔 스케줄링에 관련된 기존 연구에 대해 설명하고, III 장에서는 규칙 기반 빔 스케줄링 알고리즘, SA를 이용한 추계적(stochastic) 빔 스케줄링 알고리즘, 그리고 레이더 부하상태에 따라 두 알고리즘을 선택적으로 적용하는 전체 빔 스케줄링 알고리즘을 제안하였다. IV장에서는 시뮬레이션을 통해 그 결과를 분석하고, 마지막으로 V장에서 결론을 맺는다.

2. 레이더 빔 스케줄링 기존 연구

레이더 빔 스케줄링은 우선 순위 할당, 스케줄링 방법의 유연성, 최적화 가능성에 따라 비적응형과 적응형 크게 두 가지 형태로 구분할 수 있는데, [1]의 조사에 따르면, 비적응형 기법은 임무 우선순위가 미리 정해져 있으며, 스케줄러는 이러한 우선순위와 정해진 휴리스틱(heuristic) 규칙에 따라 자원관리를 실행하게 된다. 적응형 기법의 경우 좀 더 복잡한 과정을 이용해 우선순위를 결정하거나 최적화된 해를 얻기 위하여 스케줄링 단계에서 다양한 기법들을 활용하는 것이 특징이다.

문헌에서 주로 언급되고 있는 비적응형 기법으로는 Butler형[2]과 Orman형[3]이 있다. 두 기법 모두 빔의 고정된 우선순위가 정해져 있으며, 스케줄링은 시간축(time-line)을 따라 진행하면서 순차적인 처리과정을 통해 결정하도록 되어있다. Butler형은 요청된 임무들에 타임밸런스를 배정하여 순차적으로 비교하도록 구성되어 있다. 타임밸런스는 스케줄링마다 증가하는 값으로 임무에 대해 남은 한계시간을 나타낸다. 이 방법의 단점은 우선순위가 높은 임무(추적)가 지속적으로 많이 요청되는 경우 우선순위가 낮은 임무(탐색)가 계속 뒤로 밀려서 수행되지 않을 수 있다는 것이다. 대신에 시간축 상에 빈 공간을 남겨 두지 않으므로, 한정된 시간자원을 모두 사용한다는 이점이 있다.

Orman형은 일정 시간 간격으로 떨어진 두 개의 다른 임무의 결합(coupled-task)을 하나의 임무로 생각한다. 하나의 임무는 레이더 펄스를 보내는 시간, 대기시간, 반송 펄스를 받는 시간으로 구성되고 성능 개선을 위해 대기 시간을 활용한다. Orman형 역시 Butler형과 마찬가지로 우선순위가 높은 임무들에 의해 낮은 우선순위를 가지는 임무의 처리가 지연될 수 있다. 이는 스케줄러가 고정된 우선순위를 사용하고 가장 높은 우선순위를 가지는 임무를 항상 먼저 처리하는 특성 때문이다. [4]는 Orman형 스케줄러를 사용하면 Butler의 결과에 비해서 임무처리 요청시각 가까이에 임무들을 스케줄링 할 수 있음을 지적하고 있지만 Orman형 스케줄러는 레이더의 가용 시간을 모두 사용하지 못한다는 큰 단점이 있다. 이러한 기존의 휴리스틱을 이용한 규칙 기반 비적응형 기법은 구현이 용이하고 빠르며, 비교적 짧은 시간 안에 제한된 문제에 한해 적절한 해를 찾을 수 있다는 장점이 있다. 하지만 짧은 시간에 많은 표적을 추적하면서 동시에 새로운 표적을 탐지해야 하는 과부하(overload) 상황에서는 그 성능이 최적화됨을 보장할 수 없다는 단점이 있다.

이러한 한계를 극복하기 위해 소프트 컴퓨팅(ANN, Fuzzy)이나 전문가 시스템을 사용한 인공지능 기반 알고리즘[5,6], 은닉 마르코프 모델(HMM)이나 마르코프 결정 모델(MDP)구조 하에서 동적 프로그래밍을 통해 문제를 해결하는 방법[7], QoS(Quality of Service)를 목적함수로 두는 Q-RAM 알고리즘[8~10]등이 제안되었으며, 이러한 알고리즘은 빔 스케줄링 문제를 파형 최적화, 추적 주기 최적화 등과 결합하여 해결하고자 한다. 이러한 최적화에 기반한 적응형 기법은 학계에서는 활발히 연구중이지만 실제 레이더 시스템에 적용하여 실시간

으로 운용되기에는 알고리즘의 복잡도나 신뢰성 측면에서 아직은 기술이 성숙되어 있지 않다고 볼 수 있다.

이렇듯, 규칙 기반 알고리즘과 최적화 알고리즘은 서로 다른 장단점이 존재하며 이런 특징들은 레이더가 직면한 전장상황, 그에 따른 레이더의 부하 상황에 따라 그 성능의 차이가 확연히 달라질 수 있다는 점이 본 논문의 출발점이라 볼 수 있다. 일례로 10개 표적을 동시 추적할 수 있는 다기능 레이더에서 1개 표적을 추적하며 남는 시간에 탐지를 수행하는 저부하 상태인 경우, 굳이 복잡한 최적화 알고리즘을 적용하여 빔 스케줄링의 복잡도를 높일 필요가 없을 뿐만 아니라, 적용한다 하더라도 규칙 기반 알고리즘 대비 그 성능 차이가 거의 없을 것이라 예측할 수 있다. 반면, 수십여 개의 표적이 출현했을 경우, 레이더는 과부하 상태가 될 것이며 다양한 자원관리 기법을 동원하여 레이더의 성능을 최적화해야 하고 이런 경우 매 순간순간 스케줄링 되는 빔이 아주 중요하다. [11]에서 언급한 것처럼 과부하 상태에서의 몇 번의 잘못된 스케줄링 결과는 마치 도미노 현상처럼 전체 시스템의 성능을 크게 좌우할 수 있다.

따라서, 본 논문에서는 레이더 부하 상황별로 규칙 기반 알고리즘과 최적화 기반 알고리즘의 성능 차이를 분석하였으며, 실시간으로 변하는 레이더 부하 상태에 따라 저부하(underload) 상태에서는 규칙 기반 알고리즘을 적용하고 과부하 상태에서는 국부 탐색(local search)에 기반한 최적화 방법 중의 하나인 SA를 이용한 추계적 레이더 빔 스케줄링 알고리즘을 적용하는 방법을 제안하고자 한다.

3. 레이더 로드에 따른 빔 스케줄링 알고리즘의 선택 적용

가. 저부하 상태를 위한 규칙 기반 알고리즘

본 논문에서는 기존의 규칙 기반 알고리즘 중 자주 언급되는 Butler 알고리즘[2]을 소개, 이를 개선하여 저부하 상태를 위한 규칙 기반 빔 스케줄링 알고리즘으로 적용하고자 한다. Butler 알고리즘은 MESAR 실험용 AESA 레이더에 적용된 것으로 알려져 있다.

Butler 알고리즘에서는 타임밸런스에 기반하여 스케줄링이 이루어지는데, 타임밸런스는 스케줄링마다 증가하는 값으로 임무에 대해 남은 한계시간을 나타낸다. 즉, 음수라면 한계시각까지 남은 시간을 나타내고 양수라면 이미 한계시각을 지나쳐 지연되고 있다는 의미이다. 만약 임무가 정확히 요청된 시간 안에 처리된다면 타임밸런스는 0이 된다. 타임밸런스와 주어진 임무 우선순위에 따라 순차적인 처리과정을 반복하게 되는데, 우선순위에 입각하여 일차적으로 임무를 선택하고, 그중에 양인 타임밸런스를 가지는 임무를 선택한 후 모든 타임밸런스를 증가시켜 시간 축에서 양의 방향으로 이동하여 계속 스케줄링을 진행한다. Butler 알고리즘에서는 레이더가 처리해야 하는 일들을 “job”, “task”, “look”으로 구분하고 있다. 먼저 job은 하나의 영역을 탐색하는 일이나 특정 표적을 추적하는 것, 혹은 추적을 초기화하는 작업 단위를 의미한다. 각각의 job은 여러 개의 task로 구성되어 있다. 예를 들어, 하나의 빔 위치를 탐색한다거나, 표적 추적이나 초기화에서 한 번의 추적 정보를 갱신하는 것을 task로 정의한다. Look은 레이더에서 하나의 task에 대하여 하나 혹은 그 이상의 동기화된 레이더 송신(transmitted coherently)을 의미한다.

■ Butler 스케줄링 알고리즘(Original Butler Algorithm) :

Step 1. 먼저, 가장 높은 우선순위부터 시작한다.

Step 2. 현재 우선순위 레벨에 이미 수행되고 있는 job이 있다면, 이 job의 나머지 look들을 스케줄링하기 위해 (job을) 선택한다. 따라서 하나의 task는 같은 레벨의 다른 task들에 방해 받지 않고 순차적으로 수행된다. (6으로 간다.)

Step 3. 현재 우선순위 레벨에 수행되고 있는 job이 없다면, 가장 큰 양의 타임밸런스를 가지는 job을 선택한다. (6으로 간다.)

Step 4. 양의 타임밸런스를 가지는 job이 현재 우선 순위 레벨에 없다면, 다음 아래의 우선 순위 레벨로 내려간다. (2로 간다.)

Step 5. 만약 우선 순위 레벨의 최하위(=탐색)라면, 절대값이 가장 작은 음의 타임밸런스를 가지는 job을 선택한다. (6으로 간다.)

Step 6. 선택된 job의 다음 task(수행해야할 task)의 look 하나를 스케줄 한다.

Step 7. 스케줄 된 look이 해당 task의 마지막 look이라면 task의 look back interval만큼 타임밸런스를 감소시킨다(탐색 및 추적 초기화). 혹은 음의 look back interval만큼의 값을 가지도록 타임밸런스를 재설정한다(추적). 스케줄 된 look이 끝나는 시점으로 시간축 상에서 이동하고 그만큼(look의 수행시간 만큼) job

table에 있는 모든 job의 타임밸런스를 증가시킨다. (1로 돌아간다.)

이와 같은 알고리즘에서는 하나의 look이 스케줄 되면 해당 task의 나머지 look들은 같은 우선순위를 가지는 다른 task들에 의해 방해 받지 않고, 순차적으로 task를 마칠 때까지 수행된다. 단, 상위 레벨에 양의 타임밸런스를 가지는 job이 있다면 그 job은 하위 레벨의 태스크 중간에 삽입될 수 있다(interleaved). MESAR 알고리즘에서는 탐색 업무를 가장 낮은 우선 순위로 가정하고 있으므로, 탐색 업무의 시간 점유율과 주기(frame time)는 표적 추적 업무의 부하 정도에 따라 감소하거나 증가한다.

이처럼 [2]에서 제시된 Butler 알고리즘은 양의 타임밸런스를 갖는 빔을 스케줄링하게 되므로, 모든 빔들이 항상 요청된 시간 이후에 처리된다는 특징이 있다. 레이더의 dwell time이 길고, 짧은 시간 안에 많은 수의 추적 임무들이 요청되는 과부하 상태에서는 빔들의 지연시간이 레이더의 전체 성능을 크게 저하할 수 있는데, 요청된 시간보다 일찍 처리되는 빔을 허락함으로써 비슷한 시간에 요청되는 빔들 간의 충돌을 해소할 수 있다는 장점이 있다. 따라서 본 논문에서는 높은 우선 순위의 레벨에서 양의 타임밸런스를 갖는 대기 빔이 없을 경우 다음 우선 순위로 내려가는 것이 아니라, 요청 시간이 아직 이른 빔들 중에서도 시간 제약을 두어 스케줄링에 포함하도록 하는 방법을 제안하고자 한다. 이때 얼마나 이른 빔까지 스케줄링 범위에 포함할 것인지를 α로 결정할 수 있도록 한다. 따라서 위의 Step 4를 다음과 같이 수정하였다.

■ 수정된 Butler 스케줄링 알고리즘(Modified Butler's Algorithm) :

Step 4-1. 양의 타임밸런스를 가지는 job이 현재 우선순위 레벨에 없다면, α 이상의 음의 타임밸런스를 가지는 job을 선택한다. (6으로 간다)

Step 4-2. 양의 타임밸런스를 가지는 job도, α 이상의 음의 타임밸런스를 가지는 job도 현재 우선 순위 레벨에 없다면, 다음 아래의 우선 순위 레벨로 내려간다. (2로 간다.)

나. 과부하 상태를 위한 SA에 기반한 추계적 빔 스케줄링 알고리즘

SA는 기본적으로 국소탐색 방법을 개선하여 전역 최적화(global optimization)문제에 대한 일반적인 확률적 휴리스틱 (또는 메타 휴리스틱) 접근 방식이다. SA의 명칭과 정신은 야금학(metallurgy)에서 담금질(annealing)에서 따온 것으로, 금속을 달군 뒤에 냉각할 때 원하는 스타일의 금속을 얻기 위해서는 담금질에서 냉각 스케줄이 매우 중요하다. 담금질에서 금속 온도가 높을 때에는 원소들이 활발하게 움직이기 때문에 원소들이 제자리를 찾기 쉬워진다. 그러나 온도가 낮을 때에는 원소들의 움직임이 안정되기 때문에 원소들이 움직임이 둔화되고 안정된 상태의 결정상태가 되기 때문에 원소들의 랜덤한 이동이 어렵게 된다.

SA란, 이러한 것들을 컴퓨터를 통해 흉내 내어, 조합최적화 문제의 최적해를 얻고자 하는 방법이며 다양한 최적화 문제에 적용되어 왔다[12~14]. SA의 가장 큰 특징은 좋지 못한 해의 채택도 어느 정도의 확률로 허락하는 것인데 처음에는 나쁜 해로의 이동을 큰 확률로 주고, 점점 그 확률을 줄이게 된다. SA에서 나쁜 해에 대한 채택 여부를 결정하기 위해 온도의 개념이 필요하다. 즉 SA 알고리즘의 초기온도에서 차츰 시간이 지나면 해당온도를 조금씩 낮춰주게 되고, 온도가 높을 때는 나쁜 해로의 허락 확률이 크고, 온도가 낮을 때는 허락 확률이 낮게 된다. 이런 SA의 개념을 레이더 빔 스케줄링에 적용하기 위해 먼저 Table 1과 같은 알고리즘 인자, 변수, 해를 평가할 cost function등을 정의하였다. 다음, 이를 Table 2와 같은 SA 알고리즘에 적용하였다.

SA를 적용한 빔 스케줄링 알고리즘 제안에 있어 핵심은 스케줄링 알고리즘 변수 및 cost function을 정의하는 일이라고 볼 수 있다.

먼저, Scheduling_Time_Interval은 알고리즘의 성능을 좌우하는 중요 변수중의 하나인데, 이 변수는 SA 알고리즘 적용시 매번 스케줄링이 필요한 시점에서 평균 몇 개의 빔의 순서를 한번에 최적화하느냐를 결정한다. 한번에 최적화해야 할 빔의 개수가 많아지면 최적해로의 수렴 속도가 늦어질 것이고 한번에 최적화해야할 빔의 개수가 너무 작으면 전체적으로 최적화의 효과가 떨어지게 될 것이다. 본 논문에서는 초기해를 구하기 위해 SCS에 속한 빔들을 랜덤하게 정렬하였다. 이웃해는 현재해에서 랜덤하게 두 개의 위치를 선택하여 두 위치에 해당하는 빔의 순서를 바꾸는 것으로 정하였다. 해당해의 cost값은 해당해에서 선정한 각 빔들의 처리순서에 대해 earliness와 lateness 정도를 구하고 이를 earliness penalty score와 lateness penalty

Table 1. Algorithm parameters, variables, and cost function for SA-based beam scheduling

Algorithm parameters and variables

- Beam Definition : Beam(id, r, p, p_i, k)
  id: beam의 식별 ID
  r: 요청시간(request time), p: 처리시간(processing time, i.e. dwell time),
  p_i: 빔의 우선 순위 레벨(beam priority level)
  k : 빔의 스케줄링 순서 (SA 과정을 통해 정해짐.)
- Priority_level p_i ∈ {1(plot confirmation), 2(track initiation), 3(active track), 4(surveillance)}
  //낮은 숫자가 높은 우선순위를 가짐.
- ct : current time (스케줄링이 이루어지는 시간.)
- Scheduling_Time_Interval = OOms (SA를 통해 순서를 최적화 할 시간 구간)
  - SCS(Scheduling Candidate Set) = {beam(id, r, p, p_i, k) | r < ct + Scheduling_Time_Interval}
  - V = Sequence of beams belong to SCS (i.e. beam ∈ SCS)

Functions and Criteria

- Neighbor generation: select random two beams on V, and exchange their orders each other.
- Cost calculation
  earliness_penalty_score = [$penalty_{p\_i=1}$ $penalty_{p\_i=2}$ $penalty_{p\_i=3}$ $penalty_{p\_i=4}$]
  //차례로 p_i = 1, 2, 3, 4에 해당
  lateness_penalty_score = [$penalty_{p\_i=1}$ $penalty_{p\_i=2}$ $penalty_{p\_i=3}$ $penalty_{p\_i=4}$]
  //차례로 p_i = 1, 2, 3, 4에 해당
  For each beam(id, r, p, p_i, k) on V schedule //k = 1, 2, 3, ...|SCS|
    Let r' = estimated request time of the k_{th} beam
      $r' = ct + \sum_{m=1}^{m=k-1} p_m$, p_m is processing time of m_{th} position beam(id, r, p, p_i, m)
    Let escore = max(0, r - r'), //earliness of the kth beam,
      then, escore * earliness_penalty_score(i, p_i)
    Let lscore = max(0, r' - r) //lateness of the kth beam,
      then, lscore * lateness_penalty_score(i, p_i)
    Let pscore = escore + lscore //penalty score of the kth beam
  sum pscores of all beams on V schedule
- Stopping criteria : T is near zero or cost value is nearly unchangeable

score를 곱하여 전체를 다 더한 값으로 구하였다.

핵심 개념을 정리하면, SA를 적용한 빔 스케줄러는 매 Scheduling_Time_Interval마다 SCS를 구하고 SCS에 속한 빔들을 대상으로 earliness_penalty_score와 lateness_penalty_score를 이용해 해의 cost 값을 가장 작게 하는 즉, penalty score를 가장 작게 하는 최적의 순서를 찾는 것이다. 이때 penalty score는 빔의 우선순위와 레이더 시스템의 목적에 따라 적절하게 정의될 수 있다.

다. 레이더 부하 상태에 따른 스케줄링 알고리즘의 선택적 적용

레이더 부하에 따른 스케줄링 알고리즘은 다음 Table 3과 같다. Main 함수에서는 Scheduling_Time_Interval 간격마다 레이더 부하 상태를 모니터링하고 그 결과가 과부하로 예측되면 SA 기반 스케줄러를, 그렇지 않으면 규칙 기반 스케줄러를 호출하게 된다. 레이더 부하 상태를 예측하는 함수는 주어진 시간 구간내에서 처리되도록 기요청된 빔들의 dwell time을 합한 값을 시간 구간으로 나눈 값이 임계값 이상이면 과부하로, 그렇지 않으면 저부하 상태로 그 결과를 예측하여 리턴 한다. 앞으로 이 알고리즘을 규칙 기반 알고리즘, SA 기반 알고리즘과 구별하여 편의상 '혼합형 알고리즘'이라 칭하고자 한다.

Table 2. SA-based beam scheduling algorithm

SA_Scheduler Algorithm
Step 1. Formulation the problem parameters Step 2. Determination of the initial schedule, generate a feasible solution V; Step 3. Initialization the cooling parameters: Set the initial value of the temperature parameter, T; Set the temperature length L; Set the cooling rate F; Set the Boltzmann factor k; Step 4. Select a neighbor V' of V Let C(V') = the cost of the schedule V' Compute the move value Δ = C(V') - C(V) Step 5. If Δ <=0 accept V' as a new solution and set V = V' Else if $e^{\frac{\Delta}{kT}} > rand(0,1)$ set V = V' Else retain the current solution V Step 6. Repeat Step 4 and Step 5 during L times Step 7. Update the Temperature $T_{m+1} = F*T_m$, m = 1, 2, 3,. Step 8. Repeat Step 4 ~ Step 7 while stopping criteria is not met

Table 3. Combined scheduling algorithm

Main Function
Step 1. Set Scheduling_Time_Interval = OOms Step 2. Monitor Radar Load Status start_time = current_time, end_time = start_time + Scheduling_Time_Interval Load_Status = Radar_Road_Monitoring(start_time, end_time) Step 3. If Load_Status == Overload perform SA_Scheduler Algorithm during Scheduling_Time_Interval Else perform Modified Butler Algorithm during Scheduling_Time_Interval Step 4. Update current_time current_time = current_time + Scheduling_Time_Interval Step 5. Go to Step 2.
int Radar_Load_Monitoring(start_time, end_time)

//Calculate expected Radar load within specified time interval (start_time, end_time)
Step 1. Current priority level = the highest priority level.
load = 0;
Step 2. While current priority level
For each beam in current level
if start_time < request time of the beam < end_time
load = load + dwell time of the beam
go down to the next priority level
Step 3. If load / Scheduling_Time_Interval > β, then load_status = OVERLOAD
Else load_status = UNDERLOAD
Step 4. Return load_status.

4. 시험 결과

가. 시뮬레이션 환경

본 논문에서 제안한 알고리즘과 기존 방법 중 널리 알려지고 인용되는 Butler형 알고리즘과 비교하기 위해 [2]의 5장에서 사용된 것과 동일한 시뮬레이션 환경을 구성하였다. 레이더 임무는 두 개의 탐색 영역으로 이루어진 공간을 탐색하다가 표적이 발견되면, 순차적으로 plot confirm, track initialization, active track (track update)을 수행하는 것이다. 탐색 영역의 구분에 대한 정보들은 Table 4와 같다.

Table 4. Simulation parameters

Region	Azimuth	Elevation	Avg. dwell time(ms)	Occupancy
1	±45°	0～20°	1.42 broadside:1	20%
2	±45°	20°～50°	3.304 broadside:2	80%

Region	# of beams	Frame Time(s)	Function time(s)	Look interval(ms)
1	283	2.010s	0.402	7.102
2	329	1.359s	1.087	4.130

표적의 출현과 탐지 및 추적은 다음과 같이 가정하였다. 전체 30초의 시뮬레이션 시간에 대해 무작위로 50개의 고정된 표적을 시뮬레이션 시간의 초기 약 20초의 시간내에 생성하고, 표적이 공간상에 최초 출현한 이후에 탐색빔의 방향에 표적이 위치하면 plot confirm과 track initialization 및 active track을 수행하도록 하였다. 탐지에서 plot confirm까지의 시간 제약은 0.1초, track initialization은 0.1초 주기로 10회, active track을 위한 추적 갱신 주기는 0.2초로 설정하였다.

빔의 dwell time 은 빔 조향 각도에 따른 SNR 손실을 보상하여 표적 탐지 성능이 유지될 수 있어야 한다. 레이더 빔 조향 각도에 Ψ에 대한 감쇠된 신호대 잡음비 S/N_{Ψ}는 다음과 같다[2].

$$S/N_{\Psi} = S/N_o \cos^3\Psi$$

이에 따라 빔 조향 각도에 따른 dwell time은 위식의 역에 비례해서 증가해야 한다. 레이더의 각 탐색 영역에 대해서 function time $t_{function}$, frame time t_{frame}, look interval Δt_{look}은 탐색영역의 레이더 시간 요구 점유율 occ_d, 총 빔의 개수 N_{beam}, 레이더 정면에서의 dwell time $t_{d,0}$에 따라 아래와 같은 식으로 주어진다.

$$t_{function} = t_{d,0} N_{beam}$$
$$t_{frame} = t_{function} / occ_d$$
$$\Delta t_{look} = t_{frame} / N_{beam}$$

3절에서 제안한 알고리즘을 실제 시뮬레이션에 적용하기 위해 필요한 인자값들 및 레이더 부하 모니터링을 위한 인자값을 다음과 같이 설정하였다.

- α for Modified Butler Algorithm = 1ms
- Scheduling_Time_Interval = 10ms
- Initial temperature parameter, T = 10;
- Temperature length L = 5;
- Cooling rate F = 0.95;
- Boltzmann factor k = 1;
- Earliness_penalty_score = [10 20 50 0]
- Lateness_penalty_score = [30 10 15 0]
- Stopping criteria : T < 0.7
- Threshold for load expectation β = 0.4

앞서 언급한 것처럼 Scheduling_Time_Interval은 알고리즘의 성능을 좌우하는 중요 인자중의 하나인데, 이 인자는 SA 알고리즘 적용시 매번 스케줄링이 필요한 시점에서 평균 몇 개의 빔의 순서를 한번에 최적화하느냐를 결정한다. Table 4에서 알 수 있듯이 시뮬레이션에서 가정한 빔의 dwell time이 1～3ms 정도이므로 Scheduling_Time_Interval을 10ms로 잡을 경우 한번에 순서를 최적화해야 할 빔의 개수는 평균 5～8개 정도로 예상할 수 있다. 또한 이 주기를 통해 레이더 부하 상태를 모니터링 하게 된다.

Penalty score에 관해서 살펴보면, 요청된 시간보다 일찍 처리됨에 따른 penalty는 active track을 가장 높이 주었는데, 그 이유는 정밀 추적을 예정된 시간보다 빨리 할 경우는 전체적으로 봤을 때 필요 이상으로 추적을 많이 하게 되어 한정된 레이더의 가용 자원을 낭비하게 되기 때문이다. 요청된 시간보다 늦게 처리되는 것에 대한 penalty는 plot confirm에 가장 높이 부

과하였다. 일반적인 레이더 시스템에서 plot confirm 빔을 통해 오표적을 판단하고 오표적이 아닐 경우, confirm빔을 통해 표적의 보다 정확한 정보를 획득할 수 있게 된다. 따라서 일반적으로 탐지후 표적이 발견된 경우 plot confirm 빔의 방사까지의 시간 제약을 강제사항으로 두고 있으며, 이러한 이유로 plot confirm이 늦어지는 경우에 대한 penalty를 가장 높이 부과하였다.

Butler 알고리즘 적용시 빔은 하나의 dwell time을 갖는 것으로 정의하였으며, dwell time이 길지 않으므로 look 단위의 스케줄링은 고려하지 않았다.

나. 빔 스케줄링 성능 분석

먼저 시뮬레이션 시간동안 스케줄링을 마친 후, 빔 처리 지연도 관점에서 성능을 비교해 보았다.

1) Modified Butler Algorithm의 빔 처리 지연도

Fig. 1은 '3.가'절의 알고리즘에서 α = 1ms로 설정한 것으로 추적 관련 빔에 대해 1ms 이르게 스케줄링하는 것을 허용한 경우 스케줄링한 결과와 본래의 Butler 알고리즘을 비교한 것이다. 앞서 언급한 것처럼 Original Butler Algorithm은 모든 빔들이 항상 요청된 시간 이후에 처리된다는 특징이 있다. Original Butler Algorithm에서는 지연 없이 처리된 빔의 개수가 146개이지만, Modified Butler Algorithm을 적용한 경우, 지연 없이 처리된 빔의 개수가 333개로, 훨씬 더 많은 빔을 지연 없이 처리할 수 있게 되었다. 짧은 시간 안

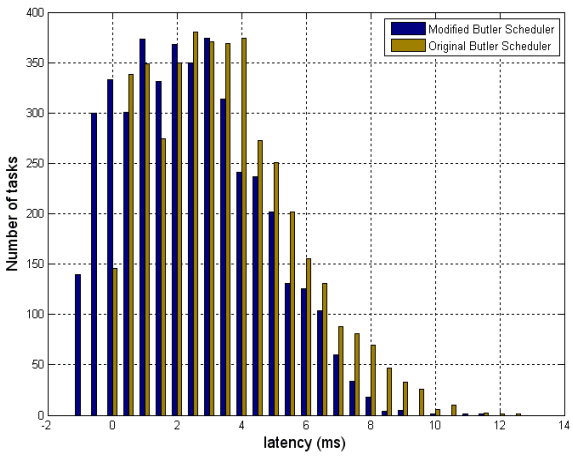

Fig. 1. Comparison between original Butler algorithm and modified Butler algorithm

에 많은 수의 추적 임무들이 요청되는 과부하 상태에서는 빔들의 지연시간이 레이더의 전체 성능을 크게 저하할 수 있는데, 요청된 시간보다 일찍 처리되는 빔을 허락함으로써 비슷한 시간에 요청되는 빔들 간의 충돌을 해소할 수 있다는 장점이 있다.

2) 규칙 기반 알고리즘, SA-기반 알고리즘, 혼합형 알고리즘 적용에 따른 빔 처리 지연도 비교

Fig. 2는 레이더 부하 상태와 관계없이 Modified Butler Algorithm으로만 스케줄링 하였을 때 과부하 상태와 저부하 상태에서의 시간 지연도 별 처리된 빔의 개수에 대한 히스토그램을 분석한 결과이다. 저부하 상태에서는 지연 없이 처리된 빔의 개수가 189개로 가장 많으며, 비교적 히스토그램의 중심이 지연시간 0 근처에 위치하고 있다. 반면 과부하 상태에서는 3ms 정도 지연된 빔들이 285개로 가장 많으며 상대적으로 2~4ms 이상 지연된 이후에 처리되는 빔들이 많고, 히스토그램의 중심도 지연시간 3ms 근처에 위치하고 있다. 이를 통해, 저부하 상태에서 과부하 상태로 갈수록 규칙 기반 알고리즘의 성능이 크게 저하됨을 알 수 있다.

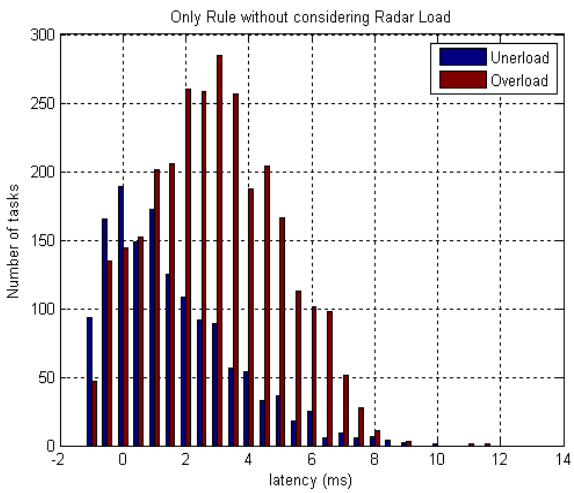

Fig. 2. Latency histogram according to load in rule-based scheduling algorithm

Fig. 3은 SA 기반 빔 스케줄링 알고리즘만으로 스케줄링 한 경우, 부하별 시간 지연도 히스토그램을 보여준다. 먼저 전체적인 그림은 저부하와 과부하 상태에서 공통적으로 지연 없이 제때 처리된 빔의 개수가

가장 많으며(저부하 상태에서 latency 0에 189개, 과부하 상태에서 latency 0에 495개) 히스토그램의 중심도 시간 지연도 0에 집중되어 있다는 것이 가장 큰 특징이다. 하지만, 규칙 기반 알고리즘보다 상대적으로 큰 지연 시간 후에 처리된 빔들, 예를 들어, 15ms 이상 지연되어 처리되는 빔들도 간혹 관찰할 수 있다. 이는 제때 처리되는 빔의 개수가 많아지는 만큼 상대적으로 많이 밀리는 빔들이 있을 수 있기 때문이다. 이를 편의상 '소수 빔의 과대 처리 지연 현상'이라 칭하도록 하자.

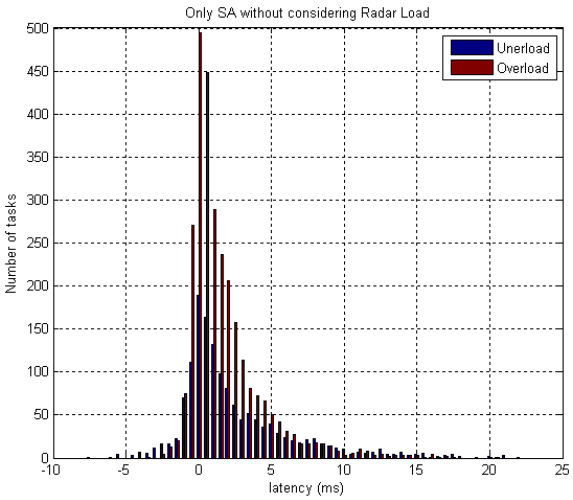

Fig. 3. Latency histogram according to load in SA-based scheduling

각 부하 상태별로 규칙 기반 알고리즘과 성능을 비교해 보면, 먼저 저부하 상태에서는 시간 지연 없이 제때 처리되는 빔들의 개수가 거의 유사하며 전반적인 지연도의 차이는 크게 없으나, 앞서 언급한 것처럼 저부하 상태에서도 SA 기반 알고리즘 적용시 '소수 빔의 과대 처리 지연 현상'이 존재한다는 단점이 있다. 반면 과부하 상태에서는 성능의 차이가 상당이 큰데, 규칙 기반 알고리즘에서는 시간 지연 없이 제때 처리되는 빔들의 개수는 144개이고 가장 많은 빔(285개)들이 3ms 정도 지연되어 처리되었다.

반면에, SA 기반 알고리즘을 적용한 경우 지연 없이 처리되는 빔의 개수가 495개로 훨씬 더 많은 빔을 제때에 처리할 수 있게 되었으며, 이후의 지연시간에서 처리되는 빔들의 개수도 급격하게 줄어든다. 이는 '소수 빔의 과대 처리 지연 현상'을 감안하더라도 과부하 상태에서는 SA 기반 스케줄링 알고리즘이 훨씬 더 효과적임을 의미한다.

Fig. 4는 '3.다'절에서 제안한 혼합형 알고리즘을 적용한 것으로, 레이더 부하에 따라 저부하 상태에서는 규칙 기반 알고리즘을, 과부하 상태에서는 SA기반 알고리즘을 선택적으로 적용한 결과이다. 저부하 상태에서 SA 기반 알고리즘 적용 대비, 규칙 기반 알고리즘을 적용함으로써 성능의 저하 없이 시스템의 복잡도를 높이지 않고도 효율적인 스케줄링이 가능한 반면, 과부하 상태에서는 규칙 기반 알고리즘 대비, SA 기반 알고리즘을 적용하여 성능이 크게 향상되었음을 동일하게 확인하였다.

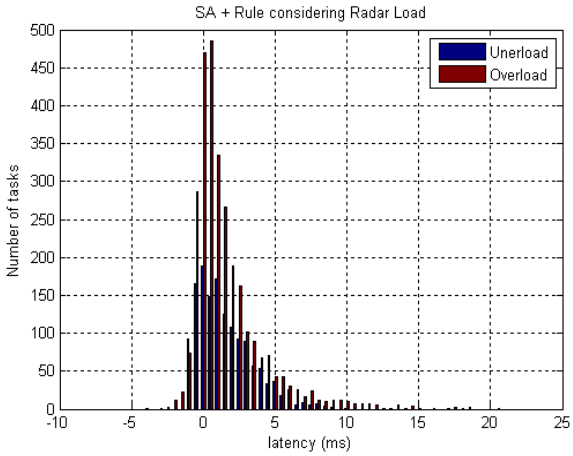

Fig. 4. Latency histogram according to load in combined scheduling algorithm

Fig. 5는 시뮬레이션 시간 동안 레이더 부하상태를 모니터링 한 것으로, 파란색 선은 '3.다'절에서 설명한 Radar_Load_Monitoring 함수에서 load 변수의 값을 그린 것이다. 빨간색 선은 레이더 로드의 변화에 따라 Radar_Load_Monitoring 함수에서 load_status의 값을 그린 것으로 이 값이 1이면 규칙 기반 알고리즘이 호출되고 2이면 SA 기반 알고리즘이 호출된다. 약 17초 기준으로 Scheduling_Time_Interval 구간 내에 기요청된 추적 관련 빔들의 수행 시간이 해당 시간 구간의 40% 이상을 차지하게 되어 레이더 부하 상태 판단을 위한 임계값 0.4를 넘어섰기 때문에 이때부터 load_status값이 2가 되어 SA기반 알고리즘이 적용되었다. 약 26초까지 레이더 로드가 증가하다가 이후는 동일한 값으로 유지되는데, 이는 26초까지 모든 표적이 다 출현하

여 추적 초기화까지 마친 상태에서, 26초 이후는 50개의 표적에 대해 정밀 추적만을 수행중이기 때문이다.

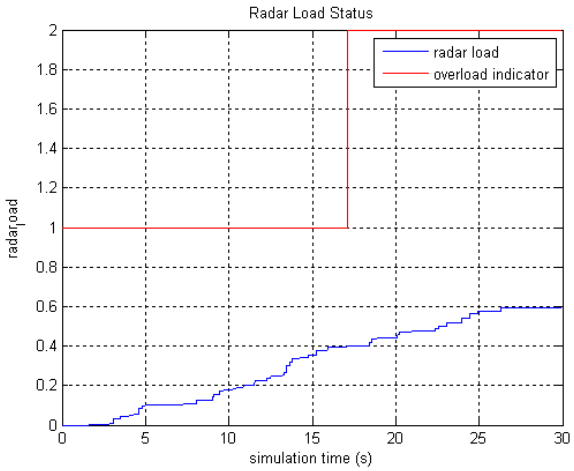

Fig. 5. Change of radar load

Fig. 6은 혼합형 알고리즘을 적용했을 때 빔별 레이더 점유율의 변화를 나타낸다. Fig. 5와 마찬가지로 약 26초 이후에 50개 표적의 정밀 추적을 60%정도로 수행하고 있으며 나머지 40%를 탐지빔에 할당하고 있고, 전 시뮬레이션 시간동안 레이더의 전체 점유율은 1로 유지되고 있어, 레이더의 idle tim은 없다.

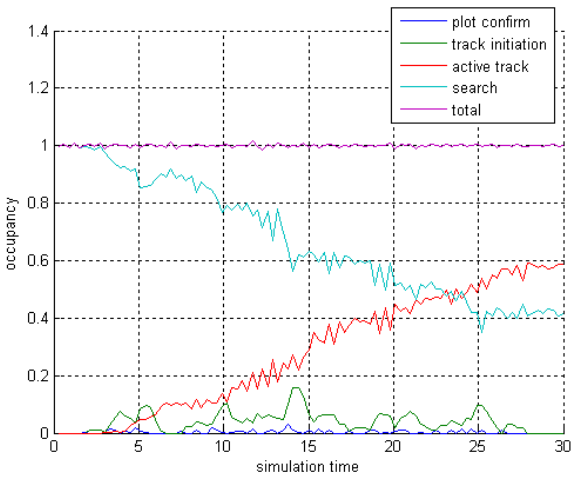

Fig. 6. Change of task occupancy

3) 스케줄링한 빔의 개수 비교

스케줄링 알고리즘의 성능 분석을 위해 처리된 빔의 개수도 중요한 성능 분석 요소 중의 하나이다. Table 5는 30초의 시뮬레이션 시간동안 표적이 50개 출현하는 동일한 시나리오에서 각 알고리즘 별 반복 실험을 통해 처리된 빔의 평균 개수를 구한 것이다. 동일한 시간 내에서 규칙 기반 빔 스케줄링보다 SA 기반 빔 스케줄링 알고리즘이 처리한 빔의 개수가 255개 더 많다. 또한 혼합형 알고리즘을 적용하는 경우도 규칙 기반 알고리즘 대비 162개의 빔을 더 처리하였다. 주어진 시간에 더 많은 빔을 처리할 수 있다는 것은, 레이더가 주어진 시간 안에 더 많은 임무를 처리할 수 있다는 뜻이며, 이는 레이더의 자원을 효율적으로 활용하기 위한 레이더 자원관리 기술의 궁극적인 목표이기도 하다.

Table 5. The number of scheduled beams in each algorithm

규칙 기반 알고리즘	SA 기반 알고리즘	혼합형 알고리즘
12,042	12,297	12,204

4) 스케줄링 수행 시간 비교

Table 6은 현재 시뮬레이션 시나리오에서 스케줄링 수행 시간과 관련된 정보를 보여준다. SA 기반 빔 스케줄링 알고리즘 적용시, Scheduling_Time_Interval을 10ms로 했을 때 SA 스케줄러가 한번에 순서를 최적화하는 빔의 개수가 평균 7.24개였으며, 이때 스케줄링에 소요되는 시간이 약 3ms 소요되었다. 즉 10ms 주기로 3ms를 소요하여 평균 7.24개의 빔을 스케줄링하게 되는 것이다.

Table 6. Scheduling time for each algorithm

		(a)	(b)	(a*b)
SA 기반 알고리즘		3ms	2,221	6.66초
규칙 기반 알고리즘		0.1ms	12,042	1.2초
혼합형 알고리즘	SA 기반	3ms	919	3.43초
	규칙 기반	0.1ms	6799	

(a) 1회 스케줄링 소요시간 (b) 스케줄러 호출 회수
(c) 전체 스케줄링 시간
수행 시간 측정 환경 : Intel(R) Core(TM)2 Quad CPU Q8300 @2.50 GHz, 3.50 GB RAM, 구현 언어: Matlab R2009b

이는 다음 스케줄링까지의 여유시간을 고려할 때, 실시간성을 보장하며 실제 레이더 시스템에 적용 가능함을 의미한다. SA 기반 알고리즘 적용시 스케줄링에 소요된 총 시간은 약 6.66초이다. 규칙 기반 알고리즘을 적용했을 경우 매번 하나씩의 빔을 스케줄링하게 되며, 한 번의 스케줄링에 소요되는 시간이 0.1ms, 스케줄링에 소요되는 총소요시간은 1.2초로 측정되었다. 혼합형 알고리즘 적용시 과부하 상태에서는 SA 기반 알고리즘이, 저부하 상태에서는 규칙 기반 알고리즘이 호출되며 전체 스케줄링 소요 시간은 약 3.4초로 SA 기반 알고리즘만 적용하는 경우 대비 약 50% 정도의 시간 절감 효과가 있다.

알고리즘의 실시간성을 판단할 때 추가적으로 고려해야 할 사항은 이러한 스케줄링 소요시간은 SA 알고리즘의 종료조건에 의해 결정되며, 이러한 종료조건은 알고리즘이 찾는 해의 최적화 정도와 trade-off관계에 있다는 것이다. 예를 들어 Scheduling_Time_Interval이 길어서 한번에 처리할 수 있는 빔의 개수가 많아질 경우 수렴 속도가 늦어져서 동일한 종료 조건을 가질 경우 해가 최적화 되지 않을 수 있고, 종료 조건을 바꿀 경우 스케줄링 소요 시간이 길어질 수 있다. 하지만 실제로 Scheduling_Time_Interval이 길어지면 오랜 시간동안의 Scheduling_Time_Interval 시간내에서 발생하는 다른 이벤트에 대한 즉각적인 반응이 어려워지는 문제가 있을 수 있다. 따라서 필요 이상으로 Scheduling_Time_Interval을 길게 설정하는 것은 수렴시간 문제를 차치하더라도 바람직하지 않다.

본 논문의 시험 결과를 통해 알 수 있듯이 SA를 이용한 스케줄링을 실제 레이더 시스템에 적용할 경우 실시간성을 보장하면서 최적화된 성능을 내기 위해서는, 해당 시스템에서 빔의 평균 dwell time을 고려하여 한번에 최적화할 빔의 개수를 평균 7개 정도로 하는 Scheduling_Time_Interval을 설정하는 것이 바람직할 것이다.

5. 결 론

본 논문에서는 레이더 자원관리의 핵심 요소라고 할 수 있는 레이더 빔 스케줄링을 위해 레이더 부하 상태에 따라 규칙 기반 알고리즘과 SA 기반 추계적 빔 스케줄링 알고리즘을 선택적으로 적용하는 것이 효과적임을 보였다. 특히, 빔의 처리시간 지연도와 스케줄링된 빔의 개수를 비교해 보았을 때 규칙 기반 알고리즘만 적용했을 때 보다 그 성능이 우월함을 입증하였다. 뿐만 아니라, 스케줄링 소요 시간 측면에서도 실시간성을 보장하는 동시에 SA 기반 스케줄링 알고리즘만 적용할 경우보다 소요 시간이 크게 단축되었음을 확인하였다. 향후 연구로는, SA 기반 스케줄링 알고리즘 적용시 나타나는 '소수 빔의 과대 처리 지연 현상'을 보완할 수 있는 방법과 추적 필터를 연동하여 표적의 추적 정확도의 측면에서 스케줄링 알고리즘을 보완하고, 다양한 성능 지수를 통해 보다 정교한 성능 평가를 수행할 필요가 있다.

References

[1] Zhen Ding, "A Survey of Radar Resource Management Algorithms", IEEE Canadian Conference on Electrical and Computer Engineering, CCECE, pp. 1559～1564, 2008.

[2] Butler, J. M., Multi-Function Radar Tracking and Control, University College London. PhD Thesis, 1998.

[3] Orman, A. J., Potts, C. N., Shahani, A. K. & Moore, A. R., "Scheduling for a Multifunction Array Radar System", European journal of Oparational Research, No. 90, pp. 13～25, 1996.

[4] Miranda, S. L. C., Baker, C. J., Woodbridge, K., and Griffiths, H. D., "Phased Array Radar Resource Management : A Comparison of Scheduling Algorithms", in Proc. IEEE Radar Conf, pp. 79～84. 2004.

[5] Komorniczak, W., Pietrasinski, J., "Selected problems of MFR Resources Management", The 3rd International Conference on Information Fusion, Vol. 2, pp. 3～8 July, 2000.

[6] Miranda, S. L. C., Baker, K., Woddbridge, K. and Griffiths, H. D., "Fuzzy Logic Approach for Prioritisation of Radar Tasks and Sectors of Surveillance in Multifunction Radar", IET Radar Sonar Navigation, Vol. 1, No. 2, pp. 131～141, 2007.

[7] Krishnamurthy, V., and Evans, R. J., "Hidden Markov Model Multiarm Bandits : A Methodology for Beam Scheduling in Multitarget Tracking", IEEE Transactions

Onsignal Processing, Vol. 49, No. 12, pp. 2893～2908, 2001.

[8] Ghosh, S., Rajkumar, R., Hansen, J., and Lehoczky, J., “Integrated QoS Aware Resource Management and Scheduling with Multiresource Constraints”, Real Time System, Vol. 33, pp. 7～46, 2006.

[9] Gopalakrishnan, S., Shih, C. S., Ganti, P., Caccamo, M., Sha, L., “Radar Dwell Scheduling with Temporal Distance and Energy Constraints”, International Radar Conference, 2004.

[10] Harada, K., Ushio, T., Nakamoto, Y., “Adaptive Resource Allocation Control for Fair QoS Management”, IEEE Transactions on Computers, Vol. 56, No. 3, pp. 344～357, 2007.

[11] Barbaresco, F. “Intelligent Multimission Radar Resources Management”, IEEE Radar Conference Tutorial Material, 2008.

[12] V. Cerny, “A Thermodynamical Approach to the Traveling Salesman Problem : An Efficient Simulated Annealing Algorithm”, Journal of Optimization Theory and Applications Vol. 45, pp. 41～51

[13] K. Dowsland, “Variants of Simulated Annealing for Practical Problem Solving”, V. Rayward-Smith Editor, Applications of Modern Heuristic Methods, Henley-on-Thames : Alfred Walter Ltd., 1995.

[14] I. H. Osman and C. N. Potts, “Simulated Annealing for Permutation Flow-Shop Scheduling”, Omega 17, pp. 551～557, 1989.