공학석사학위 논문

선입-선처리 알고리즘 기반 항공기 지상 이동 스케줄러 개발

Development of an Airport Surface Movement Scheduler Using First-Come First-Served Algorithm

2017년 2월

인하대학교 대학원

항공우주공학과(항공우주공학전공)

강 선 영

공학석사학위 논문

선입-선처리 알고리즘 기반
항공기 지상 이동 스케줄러 개발

Development of an Airport Surface Movement Scheduler Using First-Come First-Served Algorithm

2017년 2월

지도교수 이 학 태

이 논문을 석사학위 논문으로 제출함

이 논문을 강선영의 석사학위논문으로 인정함.

2017년 2월

주심 최 기 영 (인)

부심 이 학 태 (인)

위원 유 창 경 (인)

초 록

항공운송시장이 지속적으로 성장함에 따라 항공 교통량도 함께 증가하고 있다. 증가하는 교통량을 안전성을 저해하지 않고 수용하기 위해 많은 국가들은 Aviation System Block Upgrade (ASBU) 또는 Next Generation Air Transportation System (NextGen)와 같은 프로그램에 따라 항공 교통 관리 시스템에 대한 업그레이드를 진행하고 있다.

본 논문에서는 공항의 출도착 교통의 효율적인 관리를 위하여 선입-선처리(First-Come First-Served, FCFS) 알고리즘을 적용한 항공기 지상 이동 스케줄러를 개발하였다. 본 스케줄러에 이용한 선입-선처리 알고리즘은 두 단계로 구성되는데, forward propagation을 통하여 경로를 따라 모든 제약 조건을 충족하는 최단 도착 시각을 구하고 backward propagation을 통하여 출발 시각을 계산하게 된다. 개발한 스케줄러는 활주로, 유도로, 그리고 계류장을 포함하는 공항 지면의 노드-링크 구조를 사용한다. 기존에 공역 상의 항공기 교통 관리 문제를 처리하기 위해 개발되었던 선입-선처리 알고리즘을 항공기 지상 이동 스케줄링에 적용하기 위해, 링크 방향성을 처리할 수 있도록 스케줄링 알고리즘을 개선하였다. 먼저 간단한 공항 노드-링크 모델을 이용하여 개발한 스케줄러의 테스트를 진행한 후, 실제 제주 국제공항의 비행 계획을 적용하여 테스트하였다. 스케줄링 결과, 개발한 스케줄러가 주어진 경로를 따라 모든 제한 조건을 충족하는 효율적인 지상 활주 스케줄을 생성할 수 있음을 보여준다.

핵심어 : 선입-선처리 알고리즘 (FCFS algorithm), 지상 이동 스케줄링(Airport Surface Movement Scheduling), 노드-링크 모델

ABSTRACT

Air traffic volume is increasing due to continuously growing demand. To accommodate the growing traffic without compromising safety, many countries are planning to upgrade the air traffic management system under programs such as Aviation System Block Upgrade (ASBU) or Next Generation Air Transportation System (NextGen).

In this paper, an airport surface movement scheduler using First-Come First-Served (FCFS) algorithm is developed to efficiently manage surface traffic. The FCFS algorithm consists of two steps: forward propagation to find the earliest arrival time that satisfies all the constraints along the path and backward propagation to find the departure time. The scheduler uses a node-link representation of airport surface including runway, taxiway, and ramp area. To apply this FCFS algorithm that was originally developed to handle enroute traffic flow management problems to airport surface movement scheduling, the scheduling algorithm was enhanced to handle link directionality. The scheduler was initially tested with a simplified airport node link model, and then applied to Jeju International Airport with actual flight schedules. The results show that the scheduler generates efficient taxi schedule along the given route that satisfies all the capacity and junction constraints.

핵심어 : First-Come First-Served (FCFS) Algorithm,
Airport surface movement Scheduling, Node-Link Model

목 차

1. 서론 ····· 1
1.1. 연구개요 ····· 1
2. 항공기 지상 이동 스케줄링 ····· 6
2.1. 노드-링크 모델 ····· 6
2.2. 입력데이터 ····· 7
2.3. 선입-선처리 알고리즘 ····· 9
2.3.1. Forward propagation ····· 9
2.3.2. Backward Propagation ····· 10
2.3.3. State Update ····· 11
3. 스케줄링 제약조건 ····· 13
3.1. 노드 제약조건과 링크 제약조건의 차이 ····· 13
3.2. 링크의 방향성 ····· 13
3.3. 교차로 ····· 15
3.4. 링크의 최대 항공기 수용량 ····· 17
3.5. 활주로 ····· 18
4. 알고리즘 검증 ····· 19
4.1. 링크 최대 수용량 및 링크 방향성 검증 ····· 19
4.1.1. 테스트 모델 ····· 19
4.1.2. 테스트 결과 ····· 21
4.2. 교차로 제약조건 검증 ····· 23
4.2.1. 테스트 모델 ····· 23
4.2.2. 테스트 결과 ····· 24
4.3. 이착륙 간격 검증 ····· 25
4.3.1. 테스트 모델 ····· 25
4.3.2. 테스트 결과 ····· 25

5. 실 데이터를 이용한 스케줄링 수행 및 결과 ····· 27
6. 결론 ····· 33
7. 참고문헌 ····· 34

표 목차

표 1. IATA 전년 동월대비 및 누적대비 지역별 화물성장률 비교 ····· 2
표 2. 입력스케줄 ····· 7
표 3. 그림11의 링크 방향 지정 ····· 14
표 4. 노드 및 링크 구분 ····· 19
표 5. 테스트 시나리오 ····· 20
표 6. 스케줄링 결과 ····· 21
표 7. 교차로 제약조건 테스트 결과 ····· 25
표 8. 제약 조건에 따른 평균 지연 (분) ····· 29
표 9. 속도 변화 허용 시 평균 지연 시간 감소 (분) ····· 32

그림 목차

그림 1. 국제선 여객 전년 동월대비 증감률 ····· 1
그림 2. 연도별 항공기 이용객 추이(16년도는 1-7월 이용객 수) ····· 2
그림 3. Graphic depicting the ASBU Modules converging over time on their target operational concepts and performance improvements[3]. ····· 3
그림 4 우리나라의 공역[6] ····· 4
그림 5. 제주공항 노드-링크 모델 ····· 6
그림 6. 선입-선처리 알고리즘의 전체적인 과정 ····· 9
그림 7. N의 출발 가능·불가능 시간 구간과 링크 내 이동 가능 시간 구간 ····· 10
그림 8. Backward propagation ····· 11
그림 9. 시작 노드와 마지막 노드의 상태 업데이트 ····· 12
그림 10 링크의 방향성 ····· 14
그림 11. 항공기 이동 경로 도식화 ····· 14
그림 12 교차로의 노드-링크 형태 및 교차로에서의 진입상황 ····· 16
그림 13. 지상 이동 스케줄러의 선입-선처리 알고리즘 프로세스 ····· 16
그림 14. 시간에 따른 링크 내 항공기 대수 변화 ····· 18
그림 15. 검증용 테스트 노드-링크 모델 ····· 19
그림 16. 링크 L3의 시간에 따른 항공기 대수 변화(Max.5) ····· 22
그림 17. 링크 L4의 시간에 따른 항공기 대수 변화(Max.5) ····· 22
그림 18. 링크 L3의 시간에 따른 항공기 대수 변화(Max.3) ····· 23
그림 19. 교차로 제약 조건 검증용 시나리오의 이륙항공기 이동 경로 ····· 24
그림 20. 이착륙 간격 1분일 때의 링크 AA2 항공기 수 변화 ····· 26
그림 21. 이착륙 간격 2분일 때의 링크 AA2 항공기 수 변화 ····· 26
그림 22. 07번 활주로를 이용하는 경우의 출발 항공기 이동경로 ····· 27
그림 23. 25번 활주로를 이용하는 경우의 출발 항공기 이동경로 ····· 28
그림 24. 링크 P2의 항공기 수 변화 ····· 30

그림 25. 링크 P4의 항공기 수 변화 ····· 30
그림 26. 링크 AA2의 항공기 수 변화 ····· 31
그림 27. 링크 DD의 항공기 수 변화 ····· 31
그림 28. 항공기 지연 시간 히스토그램 ····· 32

1. 서론

1.1. 연구개요

항공기의 발전으로 전 세계 어떤 곳이든 비행기를 타면 자유롭게 여행을 떠날 수 있는 시대를 우리는 살고 있다. 그림1은 2012년부터 2016년까지의 8월 우리나라의 국제선 이용객 추이를[1], 그림2는 2002년부터 2016년까지의 연도별 항공기 이용객 추이를[2] 연도별로 나타낸 것이다. 그림1을 살펴보면 2012년부터 2016년까지 동월대비 국제선 국내 이용객의 수가 지속적으로 증가하는 것을 알 수 있다. 또한 그림2에서 2016년도 측정치는 1월부터 7월까지의 이용객 수를 나타낸 것으로, 16년을 제외하고 2007년 이후 약간의 감소가 있었지만 지난 13년 동안 꾸준히 증가하고 있는 추세이다. 표1은 국제항공운송협회(International Air Transport Association, IATA)에서 측정한 전년 동월대비 및 누적대비 지역별 화물성장률을 비교한 것이다[1]. 표1의 FTK(화물 톤킬로미터)는 항공화물 운송실적을 산출하는 기준으로 15년 7월 대비 16년 7월에 5% 증가한 것을 볼 수 있다. 그림1과 그림2, 그리고 표1에서 볼 수 있듯이 항공운송시장은 지속적으로 성장하고 있다.

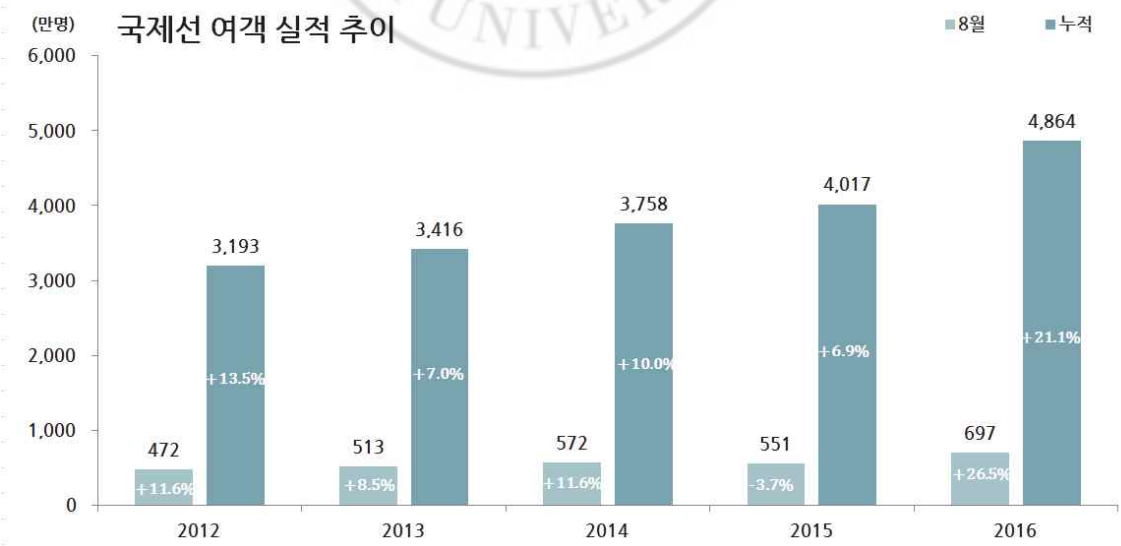

그림 1. 국제선 여객 전년 동월대비 증감률

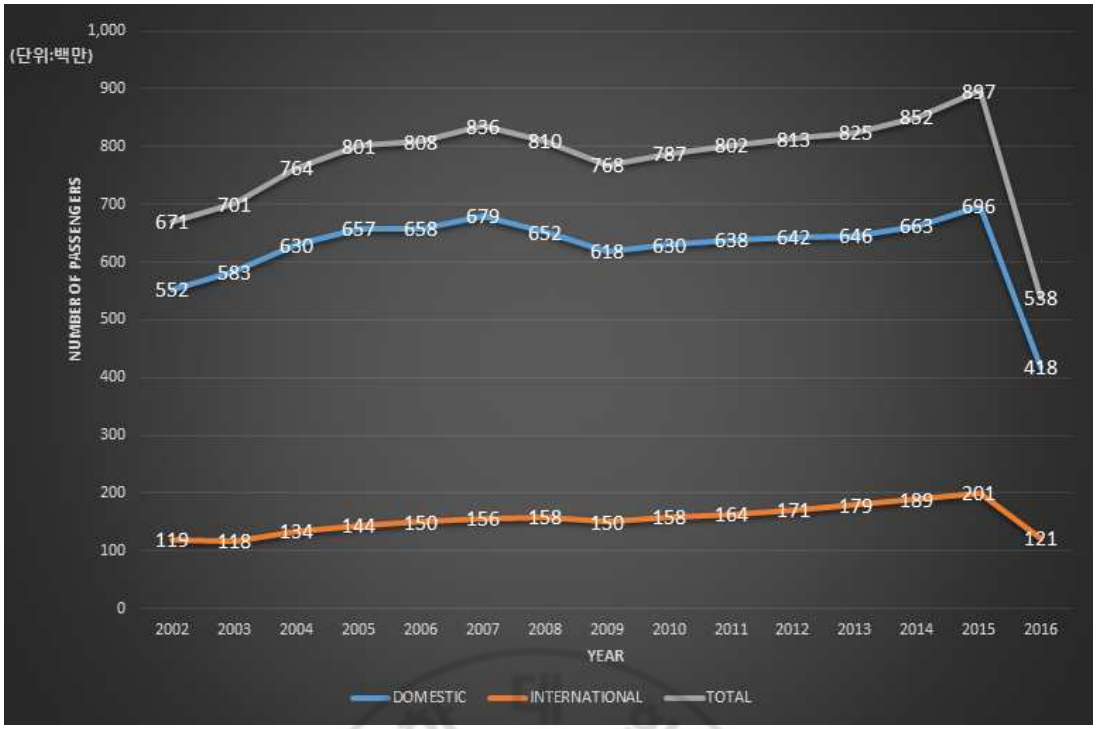

그림 2. 연도별 항공기 이용객 추이(16년도는 1-7월 이용객 수)

표 1. IATA 전년 동월대비 및 누적대비 지역별 화물성장률 비교

(단위: %)

구분	FTK 시장 점유율	전년 동월대비 (2015.7월 vs 2016.7월)			
		FTK	AFTK	FLF(%p)	FLF
아시아태평양	38.9	4.9	2.7	1.1	53.1
유럽	22.3	7.2	3.8	1.4	43.2
북미	20.5	4.1	3.4	0.2	31.7
남미	2.8	-5.6	10.1	-5.0	30.0
중동	14.0	6.7	11.0	-1.6	39.2
아프리카	1.5	-6.8	31.3	-7.9	19.4
전체	100.0	5.0	5.2	-0.1	41.3
구분	FTK 시장 점유율	누적대비 (2015.1~7월 vs 2016.1~7월)			
		FTK	AFTK	FLF(%p)	FLF
아시아태평양	38.9	-1.2	3.6	-2.5	51.6
유럽	22.3	4.4	5.9	-0.6	44.5
북미	20.5	-0.1	4.8	-1.6	33.1
남미	2.8	-5.1	2.5	-2.7	34.3
중동	14.0	6.5	10.6	-1.6	41.0
아프리카	1.5	-2.7	24.2	-6.6	23.7
전체	100.0	1.2	5.8	-1.9	42.3

FTK: 화물 톤킬로미터(수송톤수x비행거리)
AFTK: 공급 화물 톤킬로미터(공급 화물 탑재중량x비행거리)
FLF: 화물 탑재율(공급 화물 톤킬로미터에 대한 화물 톤킬로미터의 비율)

이와 같은 항공운송시장의 성장과 함께 항공교통량이 지속적으로 증가하면서 업계에서는 현재 운항 시스템의 한계를 느끼고 있으며, 이러한 문제를 극복하기 위한 대안으로 국제민간항공기구(International Civil Aviation Organization, ICAO)에서는 Aviation System Block Upgrade(ASBU)를, 미국 연방항공청(Federal Aviation Administration, FAA)에서는 Next Generation Air Transportation System(NextGen) 프로젝트를 추진하여 새로운 항행 시스템을 개발하기 위해 노력하고 있다. 그림3은 ASBU가 포함하고 있는 항목과 block별 각 항목의 capability를 모듈로 도식화한 것이다.

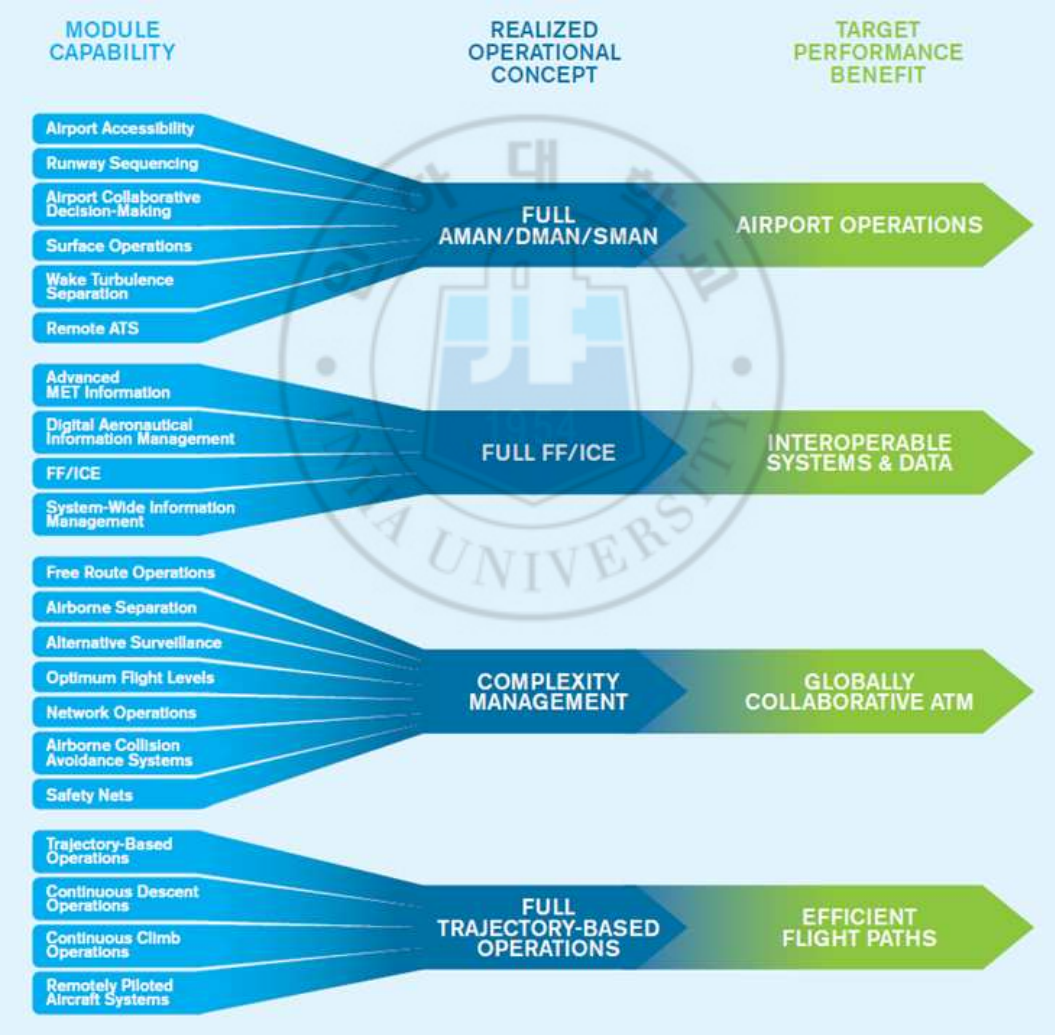

그림 3. Graphic depicting the ASBU Modules converging over time on their target operational concepts and performance improvements[3].

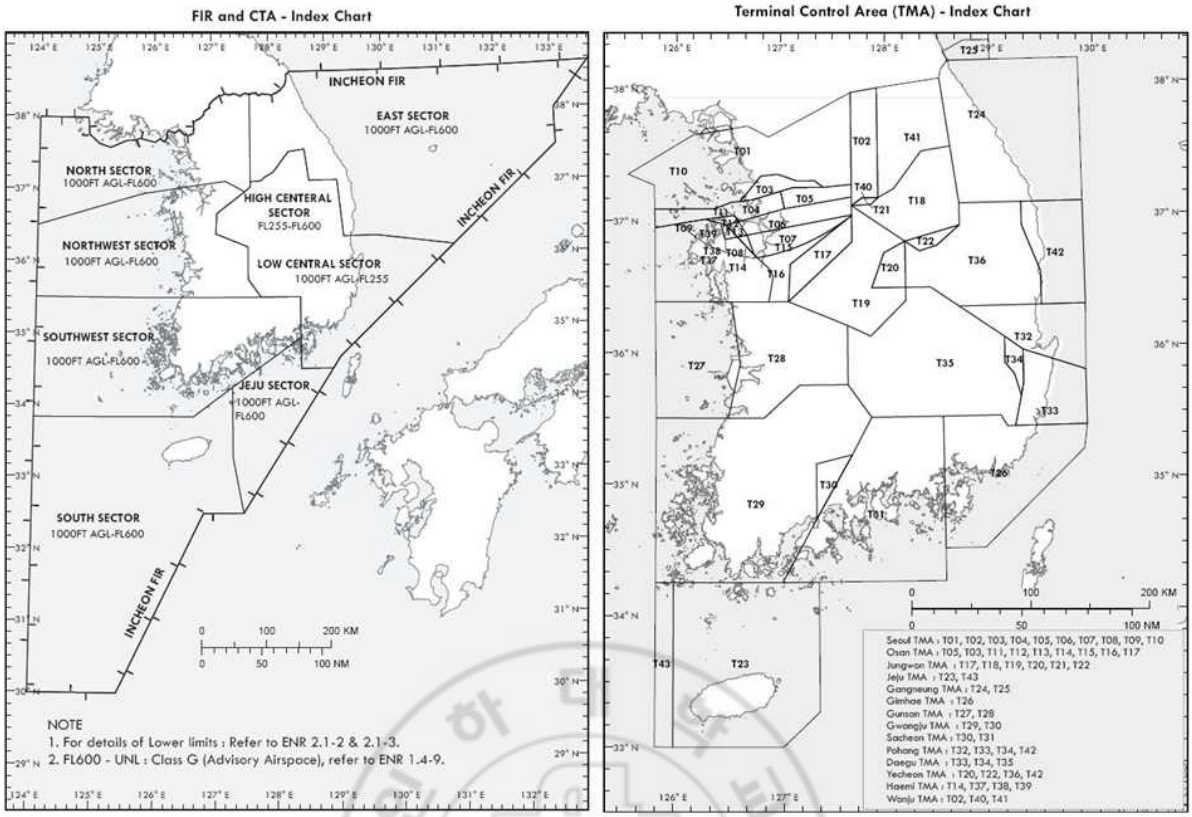

그림 4 우리나라의 공역[6]

그림3을 살펴보면 ASBU에서 진행하고 있는 차세대 항행 시스템 항목 중에 항공기의 운항 지연을 줄이기 위한 항공기 출·도착 관리(DMAN, AMAN)와 surface operation에 대한 연구가 포함되어 있다. 항공기 출·도착 지연을 줄이기 위해서 미국항공우주국 (National Aeronautics and Space Administration, NASA)의 Ames Research Center에서는 선입-선처리 (First-Come First-Served, FCFS) 알고리즘을 기초로 한 스케줄링 기법에 대해 연구를 진행해왔다[4]. 해당 연구는 출발공항부터 도착공항까지의 운항 궤적을 노드-링크 구조(Node-Link Structure)로 간소화하여, 출발 및 도착공항과 공역 사이의 경계를 노드로 공역 구간은 링크로 간주하였다. 항공기별 예정된 스케줄을 링크와 노드의 제약조건을 만족하도록 다시 스케줄링하여 항공기의 운항 중 지연을 줄일 수 있는 방법을 제안하였다. 인하대학교에서는 FCFS 알고리즘 기반의 스케줄링 기법을 그림4와 같은 우리나라 공역에 적용시키기

위한 연구를 진행했었으며, 최종적으로 다중 FCFS MP스케줄링 소프트웨어를 개발하였다[5].

본 논문에서는 NASA에서 제안한 선입-선처리 알고리즘 기반 스케줄링 기법을 항공기 지상 스케줄링에 도입하여 공항 내에서의 지연을 줄일 수 있도록 하는 스케줄러 개발에 대해 연구한다. 스케줄링에 필요한 기본적인 틀은 앞서 개발한 다중 FCFS MP스케줄링 소프트웨어와 동일하게 제작하였으며, 지상 이동 스케줄링에 필요한 부분을 더 추가하고 개선한다.

2. 항공기 지상 이동 스케줄링

2.1. 노드-링크 모델

항공기 지상 이동 스케줄링을 위해 출발 및 도착하는 항공기가 공항 내에서 이동하는 경로를 노드-링크 모델로 적용시킨다. 공항내의 항공기 이동로는 활주로(Runway), 유도로(Taxiway), 그리고 계류장(Ramp)이 있다. 공항에서 출발(이륙)하는 항공기는 일반적으로 하나의 게이트로부터 활주로까지 이동하며, 도착(착륙) 항공기의 경우는 반대로 활주로에서 게이트로 이동한다. 따라서 공항의 노드-링크 모델은 게이트부터 활주로까지의 항공기가 이동하는 모든 경로에 대해 모델링을 하며, 계류장에서의 항공기 이동은 게이트에서 유도로까지의 직선 경로로 간주한다.

그림5은 하늘에서 바라본 제주공항의 전경으로 제주공항을 대략적인 노드-링크 모델로 나타낸 것이다. 그림에서 알 수 있듯이 게이트는 노드로 나타내었으며 교차로 또한 노드로 표현하였다.

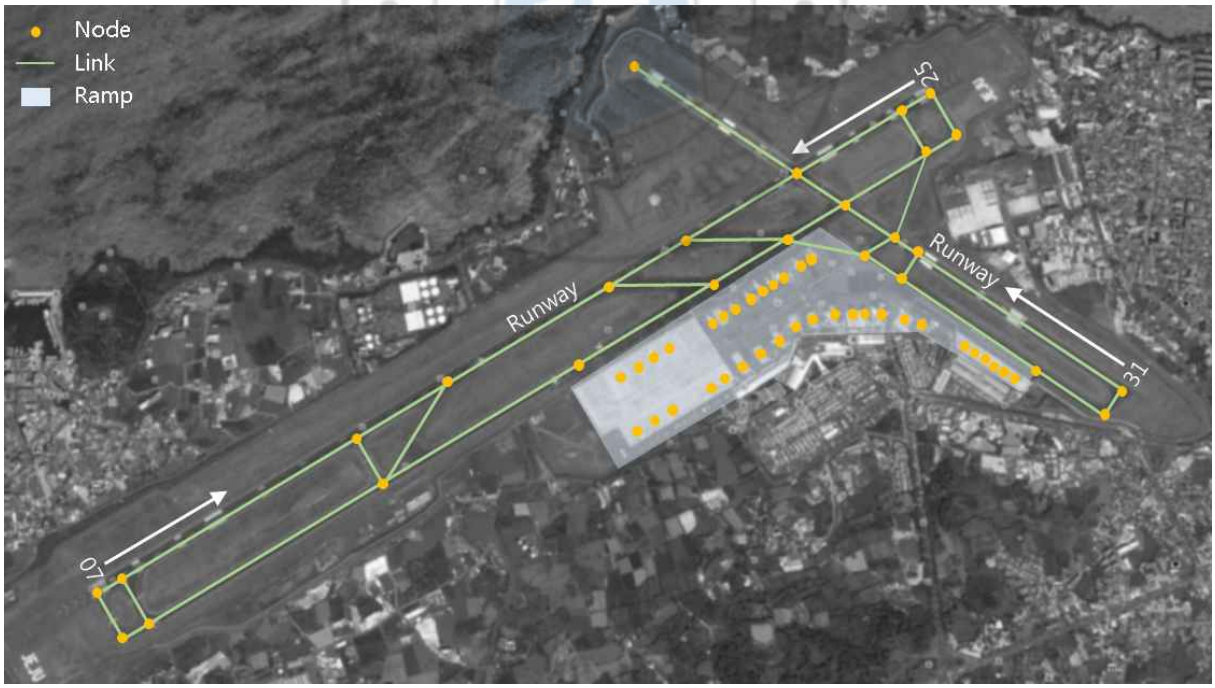

그림 5. 제주공항 노드-링크 모델

2.2. 입력데이터

선입-선처리 알고리즘을 기반으로한 항공기 지상 이동 스케줄링에는 기본적으로 몇 가지의 입력데이터를 필요로 한다. 먼저 각 노드와 링크의 기본적인 정보가 필요하다. 이는 노드의 위도·경도와 각 노드·링크가 활주로의 한 부분인지, 유도로의 한 부분인지, 또는 게이트인지 등의 내용을 포함한다. 또한 지상 이동 스케줄링에 있어서 링크의 방향성을 판단하기 위해 링크별 노드 정보가 필요하기 때문에 각 링크를 구성하는 노드의 정보도 포함된다. 링크의 방향성에 대한 내용은 뒤에서 자세하게 설명하도록 하겠다. 스케줄링에 필요한 또 다른 입력데이터는 해당 공항의 시간별 Aircraft Departure Rate(ADR)와 Aircraft Arrival Rate(AAR) 정보와 게이트에서의 turnaround time, 그리고 링크별 시간에 따른 최대 수용량 정보를 요구한다.

표 2. 입력스케줄

Time unit: sec

Flight ID	Entry Time	Exit Time	Transit Time	Previous Link	Current Link	Next Link	Speed-up (%)	Slow-down (%)
F01	29100	29127	27	XXXX	**N01**	L01	1	1
F01	29127	29140	13	L01	L02	L03	0	2
F01	29140	29181	41	L02	L03	L04	0	2
F01	29181	29195	14	L03	L04	L05	1	1
F01	29195	29218	23	L05	**N02**	XXXX	0	1
F02	29100	29127	27	XXXX	**N10**	L10	0	3
F02	29127	29140	13	L10	L20	L30	1	3
F02	29140	29181	41	L20	L30	L40	1	2
F02	29181	29195	14	L30	L40	L50	0	2
F02	29195	29218	23	L50	**N20**	XXXX	1	2

본 논문에서 스케줄링에 이용한 선입-선처리 알고리즘은 노드-링크 모델로 적용한 항공기의 이동경로를 이용하여 스케줄링하기 때문에 정형화된 입력스케줄이 필요하다. 요구되는 입력스케줄의 형태는 표2와 같다. ‘Flight ID’는 말 그대로 항공기의 ID 또는 callsign을 나타내고, ‘Previous Link’, ‘Current Link’, ‘Next Link’는 각각 항공기가 바로 전에 지나온 링크, 현재 위치한 링크, 다음으로 진입할 링크이다. 항공기

지상 스케줄링에서 출발하는 항공기의 경우는 앞에서 설명한 바와 같이 게이트에서 활주로로 이동하게 되므로 첫 번째 current link는 해당 항공기가 정차되어 있던 게이트가 되며, 마지막 current link는 활주로 위의 노드가 된다. 반대로 도착하는 항공기의 첫 번째 current link는 활주로 위의 노드가 되고, 마지막 current link는 게이트가 될 것이다. ‘Entry Time’, ‘Exit Time’, ‘Transit Time’은 각각 현재 링크에 들어가는 시간, 나가는 시간, 링크 내 이동시간을 나타내며, 단위는 초단위이다. 표2에서 음영처리한 각 항공기별 첫 번째 entry time과 마지막 exit time은 게이트 또는 활주로에의 예정된 출발 시간 또는 도착시간을 나타낸다. 만약 F01 항공기가 출발하는 항공기인 경우, ‘29100’은 항공기가 게이트를 출발하는 시간이 되고, ‘29218’은 항공기가 활주로에 도착하는 시간이 된다. ‘speed-Up’과 ‘slow-Down’은 항공기 속도 제한을 퍼센트로 나타낸 것이다.

2.3. 선입-선처리 알고리즘

본 논문에서 스케줄링에 이용한 선입-선처리 알고리즘은 크게 earliest arrival time을 계산하는 'Forward propagation'와 그로부터 earliest departure time을 역으로 계산하는 'Backward propagation' 두 단계로 수행된다. 알고리즘의 전체적인 과정은 그림6과 같이 나타낼 수 있으며, 알고리즘은 시간을 기본 단위로 한다.

2.3.1. Forward propagation

알고리즘 수행에 있어 스케줄링의 우선순위를 선정해줘야 하는데, 여기서는 예정된 출발시간(출발 항공기의 경우는 게이트 출발 시간, 도착 항공기의 경우는 활주로 착륙 시간)을 기준으로 우선순위를 부여하였다.

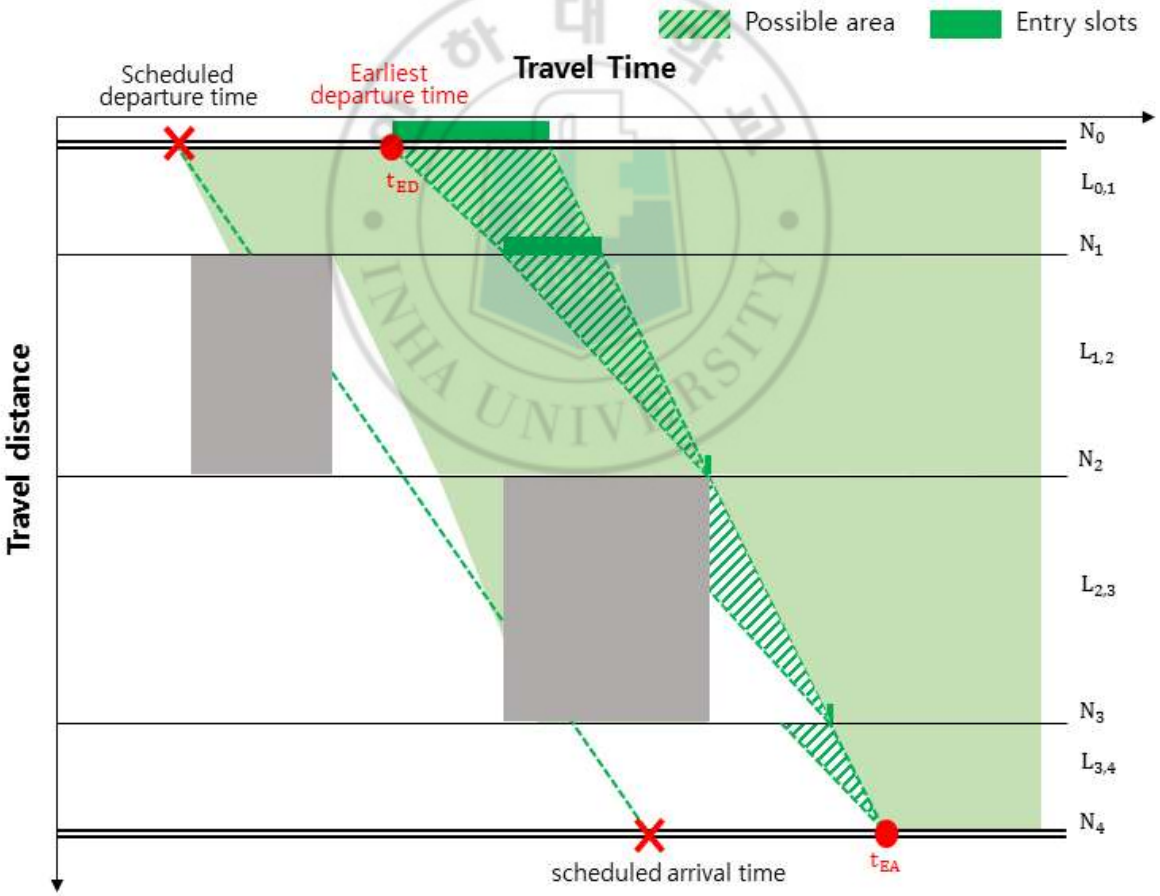

그림 6. 선입-선처리 알고리즘의 전체적인 과정

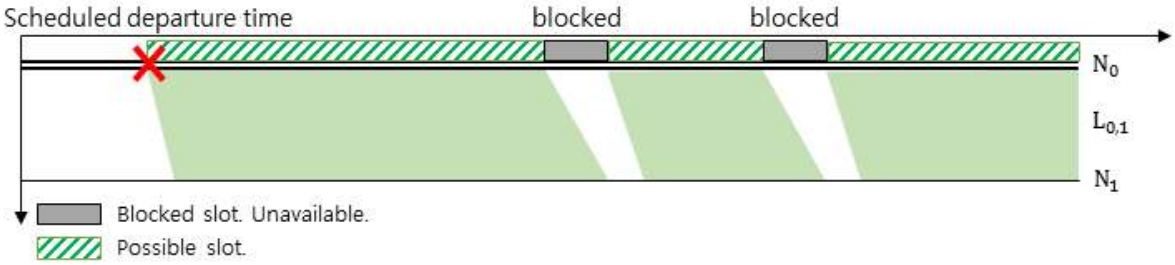

그림 7. N_0의 출발 가능·불가능 시간 구간과 링크 내 이동 가능 시간 구간

출발 노드에서 예정된 출발 시간을 기준으로 항공기가 출발 가능한 시간 구간을 먼저 찾는다. 그림7은 스케줄의 시작 지점인 노드 N_0에 대해 항공기가 출발 가능한 시간과 불가능한 시간을 나타낸 것이다. Blocked된 시간 구간에 대해 보여주기 위해 그림6과는 다르게 도시하였다. 그림7에서 blocked된 구간은 다른 항공기가 이미 배정되었거나 임의의 사유로 사용할 수 없는 구간이다. 이 시간을 제외한 나머지 구간에서는 항공기가 출발 가능하다. 이 출발 가능 시간 구간에 대해 항공기의 이동속도를 고려하여 다음 노드까지의 링크 통과 예정 구간을 찾아준다.

앞과 같은 과정을 반복하여 마지막 노드(그림5의 N_4)까지의 forward propagation을 수행한다. 그림6을 살펴보면 링크에도 block된 구간을 찾을 수 있는데, 이는 링크가 수용할 수 있는 최대 항공기 수에 도달한 경우로 해당 시간동안에는 다른 항공기가 진입할 수 없다. 마지막 노드까지 forward propagation을 모두 수행하면 earliest arrival time이 산출된다. 여기서 말하는 earliest arrival time은 propagation을 통해 계산된 마지막 노드의 항공기 도착 예정 시간 구간 중 가장 빠른 시간을 의미한다.

2.3.2. Backward Propagation

Backward propagation은 forward propagation을 통해 결정된 earliest arrival time을 기준으로 수행된다. 그림8는 N_4에서 N_3로 거슬러 올라가는 과정을 보여준다. 그림8에서 빗금 쳐진 구간은 backward propagation을 통해 구해진 이동 가능 구간이며, 단색으로 칠해진 구간

이 forward propagation을 통해 구한 가능 구간이다. Forward와 backward를 통해 구해진 두 구간이 겹치는 부분이 그림8에서 화살표로 표시한 possible slot이며, 항공기는 이 영역 내에서 자유롭게 운용될 수 있다. 그림 8의 과정을 N_0까지 반복하면 앞의 그림 6과 같은 결과를 얻을 수 있으며, Backward propagation을 마치고 나면 첫 번째 노드에서의 가장 빠른 출발 가능 시간이 계산된다.

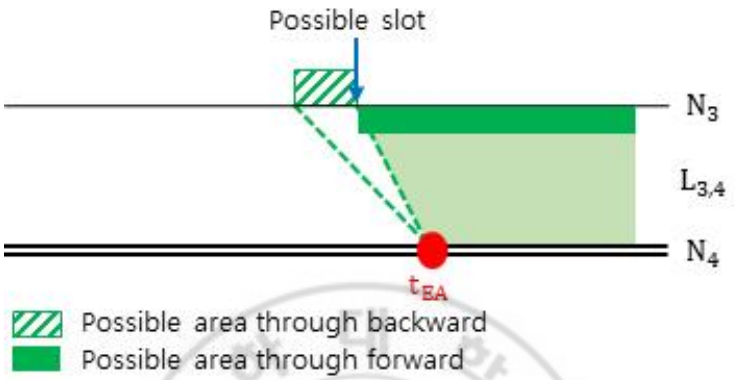

그림 8. Backward propagation

2.3.3. State Update

Forward와 backward propagation을 수행하여 항공기의 새로운 스케줄이 정해지면, 정해진 스케줄에 대해 각 링크와 노드의 상태를 갱신시켜줘야 한다. 만약 노드와 링크의 상태 갱신을 바로 해주지 않으면 다음 항공기 스케줄링에서 설정해준 제약조건을 만족하지 못하는 경우가 생기게 되므로, 한 항공기의 propagation이 끝날 때마다 링크와 노드의 상태를 업데이트 시켜준다. 노드 제약조건과 링크 제약조건의 기준이 다르며, 이에 대한 설명은 3장에서 자세하게 다룬다.

먼저 항공기의 출발 노드와 도착 노드의 상태를 업데이트 시켜준다. 스케줄의 시작과 끝 노드에서, earliest departure & arrival time을 중심으로 그림 9와 같이 closed slot을 생성한다. 이때 노드가 활주로 위의 점이라면 1시간을 공항의 시간당 ADR과 AAR로 나눈 값으로, 게이트라면 지정된 turnaround time을 기준으로 'dt'를 선정한다.

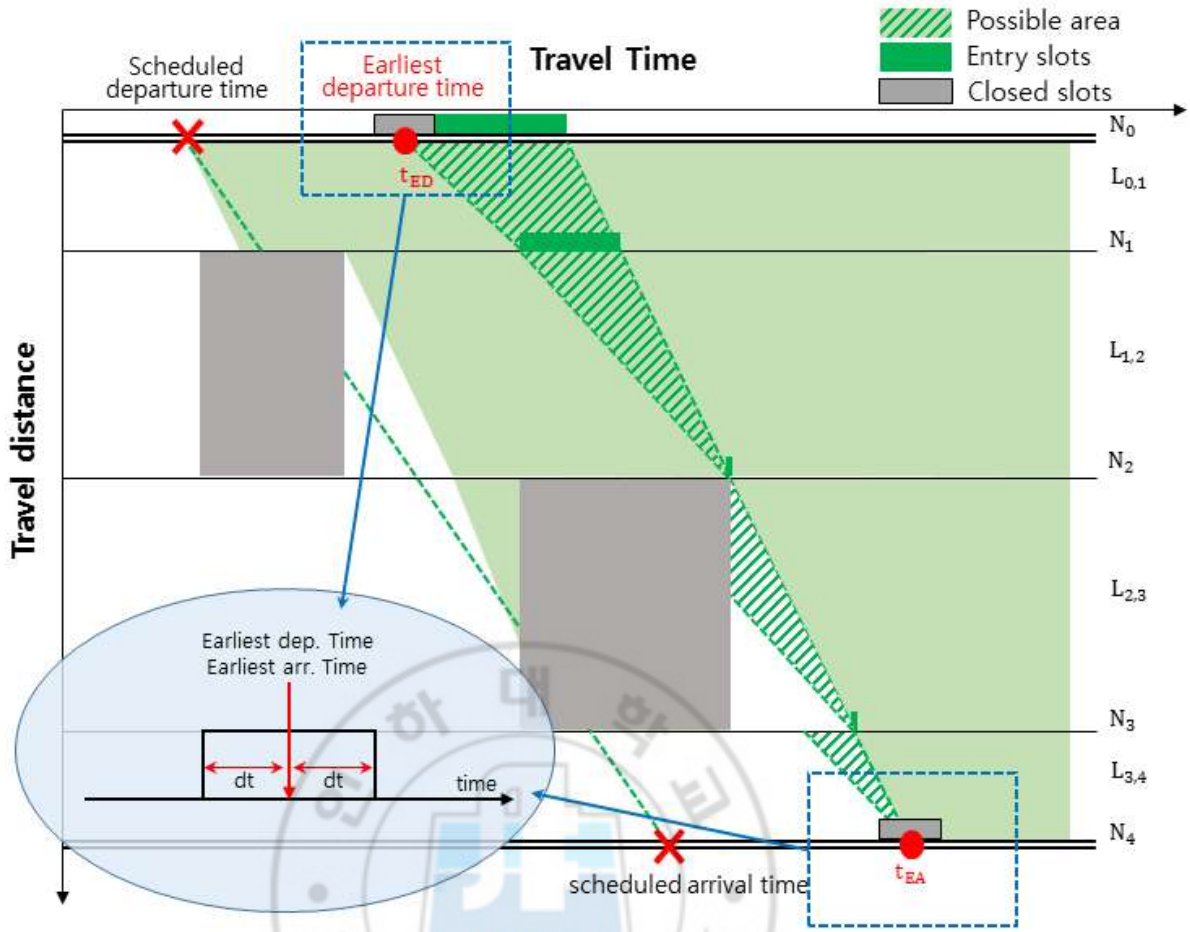

그림 9. 시작 노드와 마지막 노드의 상태 업데이트

다음으로 항공기가 통과하는 모든 링크의 상태를 업데이트한다. 링크의 경우, 항공기가 해당 링크를 통과하는 시간 구간에 대해 링크 내 항공기 수를 증가시켜줘야 한다. 링크 내 항공기 수는 링크의 최대 수용량을 초과할 수 없으며, 만약 항공기 수가 임의의 시간 간격동안 최대 수용량에 미치는 경우에는 해당 시간 간격을 닫아주어(closed) 다른 항공기가 링크에 진입할 수 없도록 한다. 그림 9에서 링크 내에 closed slots으로 표시된 부분이 링크 내 항공기 수가 최대 수용량과 같아진 경우이다.

이와 같이 노드와 링크의 상태 갱신까지 마치면 한 항공기의 스케줄링 알고리즘이 끝나게 된다. 이를 모든 항공기에 대하여 반복 수행한다.

3. 스케줄링 제약조건

항공기 지상 이동 스케줄링에는 여러 제약조건이 필요하다. 제약조건은 크게 링크의 방향성, 교차로 처리, 링크의 최대 수용량 등이 있다.

3.1. 노드 제약조건과 링크 제약조건의 차이

본 연구에서 사용하는 선입-선처리 알고리즘 기반 스케줄링에는 노드와 링크에 대한 제약조건이 필요하다. 노드의 제약조건과 링크의 제약조건은 앞에서 언급했듯이 차이가 있다.

노드 제약조건은 임의의 기준 시간에 대한 rate를 기준으로 제한된다. 이는 2.3.3절에서 언급한 공항의 ADR, AAR과 같은 형태라 얘기할 수 있으며, 기준 시간에 대한 rate를 넘지 않도록 기준시간을 rate 값으로 나눈 시간간격을 노드의 closed slot 크기로 지정한다.

링크 제약조건은 노드와 달리 rate를 제한으로 주는 것이 아닌 시간별 링크의 상태에 따라 제한한다. 링크의 항공기 최대 수용 대수를 지정하여 모든 시간 간격에서 이를 만족하도록 하며, 만약 링크 내의 항공기 수가 이 최대 수용량에 도달하면 다른 항공기가 진입할 수 없도록 해당 시간구간을 닫아준다.

3.2. 링크의 방향성

공항 내에서 항공기가 이동하는 활주로, 유도로 등의 경우, 자동차가 다니는 도로와 달리 양방향 통행이 가능하지 않다. 그림10은 항공기들의 링크 진입 상황을 나타낸 것이다. 그림10의 1번 항공기는 이미 링크에 진입하여 (+)방향으로 이동하고 있는 중이며, 2번과 3번 항공기는 링크에 진입하려 하는 상황이다. 2번 항공기의 경우는 (-)방향으로 링크에 진입하려 하지만, 이미 링크 내에서 (+)방향으로 1번 항공기가 이동하고 있기 때문에 1번 항공기가 링크를 빠져나오기 전까지 해당 링크에 진입할 수 없다. 반면에 3번 항공기는 1번 항공기와 같은 (+)방향으로 진행

하기 때문에 분리 간격만 유지한다면 바로 해당 링크로의 진입이 가능하다. 이처럼 항공기 지상 이동 스케줄링에는 모든 링크에 대한 방향성 판단이 매우 중요하다.

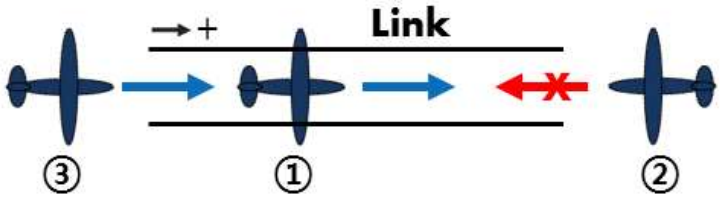

그림 10 링크의 방향성

방향성 판단을 위해 링크는 2개의 노드로 이루어진다는 점과 연결된 두 링크는 하나의 노드를 공유한다는 점을 이용하였다.

스케줄링에 필요한 입력데이터 중 링크 정보를 읽어올 때, 링크를 구성하는 두 개의 노드를 임의로 1번 노드와 2번 노드로 지정해준다. 그리고 링크의 방향을 1번 노드에서 2번 노드로 진행하는 경우는 양방향으로, 반대의 경우는 역방향으로 설정한다. 좀 더 이해하기 쉽도록 표3을 첨부한다.

표 3. 그림11의 링크 방향 지정

링크	1번 노드	2번 노드	정방향(+)	역방향(-)
Link1	Node2	Node1	Node2→Node1	Node1→Node2
Link2	Node2	Node3	Node2→Node3	Node3→Node2
Link3	Node3	Node4	Node3→Node4	Node4→Node3
Link4	Node5	Node4	Node5→Node4	Node4→Node5

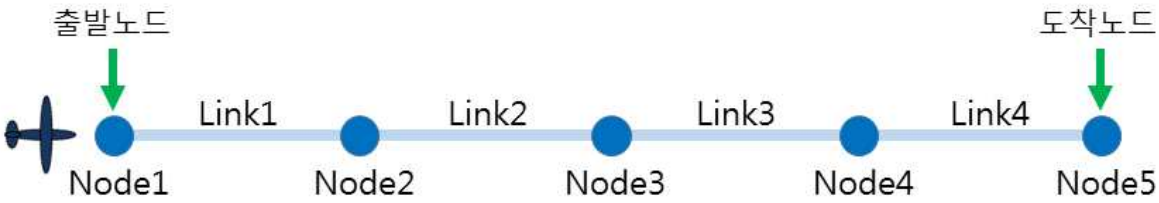

그림 11. 항공기 이동 경로 도식화

그림11에서 항공기는 Node1을 기점으로 Link1, Link2, Link3, 그리고 Link4를 거쳐 Node5에 도착하는 경로를 가지고 있다. 표2의 입력스케줄에서도 알 수 있듯이 본 논문의 스케줄링에 사용되는 모든 항공기의 스케줄은 노드로 시작하여 여러 링크를 거쳐 노드로 끝난다. 시작노드(그림11의 Node1)는 항공기가 처음으로 진입하게 되는 링크 (그림11의 Link1)를 구성하는 노드로, 항공기가 Link1로 진입하는 입구 노드라고 할 수 있다. 따라서 이 입구 노드를 링크 정보를 읽어오면서 임의로 지정해준 Link1의 1번 노드와 비교하여 두 개의 노드가 같다면 항공기는 정방향으로, 다르다면 역방향으로 진행한다는 것을 알 수 있다. 표3에서 Link1의 1번 노드는 Node2로 지정하였기 때문에 그림11의 항공기는 Link1을 역방향으로 통과하는 것이 된다.

Link1을 들어가는 노드를 알고 있으므로 Link1을 빠져나갈 때의 노드(Node2)도 알 수 있다. 이 노드는 결국 다음 링크(Link2)로 진입하는 노드가 되며, 이 노드를 Link2의 1번 노드와 비교하여 방향성을 판단하면 된다. 위 과정을 마지막 링크까지 반복하면, 출발노드부터 도착노드까지의 모든 링크의 방향성 판단이 가능해진다.

3.3. 교차로

지상 노드-링크 모델에서 교차로는 그림12과 같이 노드로 표현된다. 교차로의 경우 만약 한 항공기가 그림12과 같이 교차로 내에서 진행 중이라면, 다른 항공기는 진행 중인 항공기가 교차로를 빠져나갈 때까지 해당 교차로에 진입할 수 없다. 즉, 링크 제약조건으로 얘기하면 교차로의 최대 항공기 수용량은 '1'이라고 얘기할 수 있다.

하지만 여기서 교차로는 링크가 아닌 노드이기 때문에 앞에서 언급한 기준 시간에 대한 rate로 제약조건을 준다. 교차로의 경우 활주로에서의 AAR이나 ADR 같은 기준이 주어지는 것이 아니기 때문에 항공기가 교차로를 빠져나가는 시간을 임의로 정해준 뒤, 이 값으로 기준시간을 나눗셈하여 rate를 구해준다.

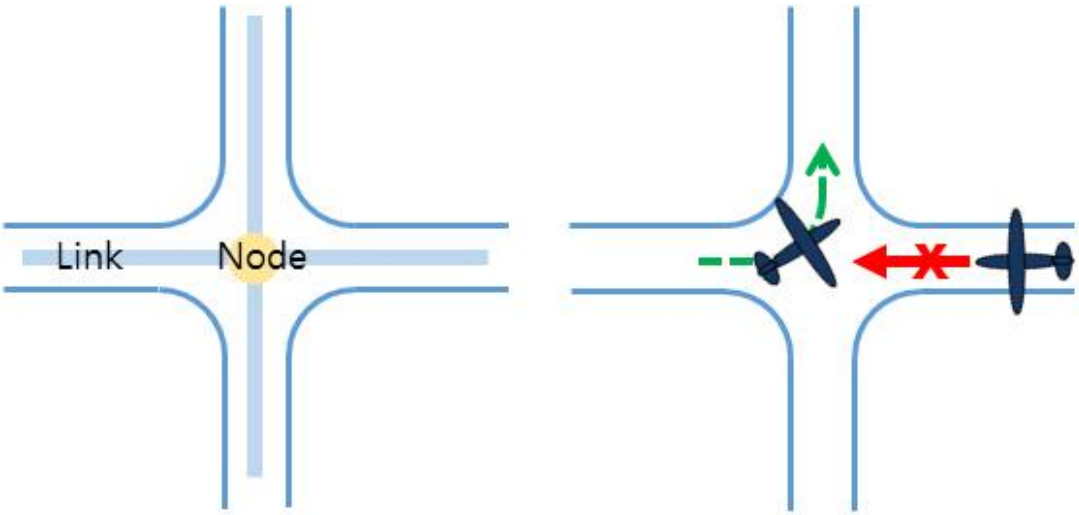

그림 12 교차로의 노드-링크 형태 및 교차로에서의 진입상황

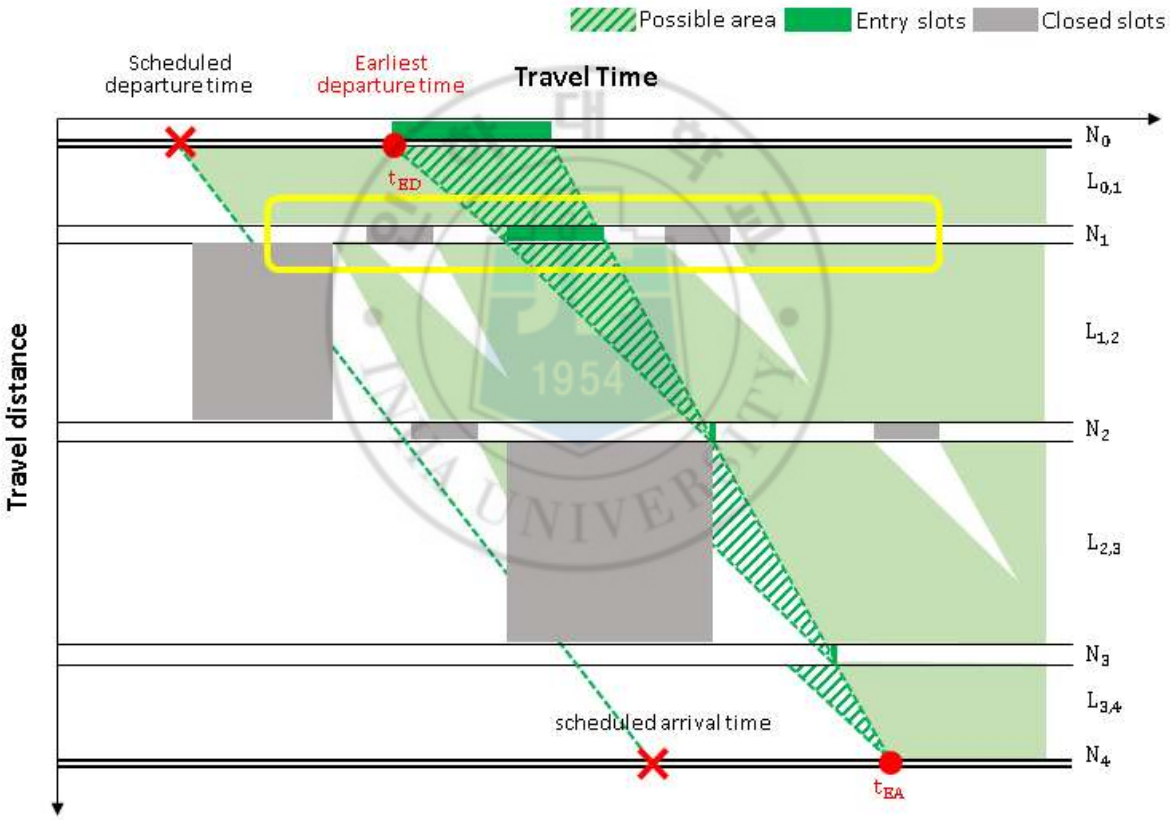

그림 13. 지상 이동 스케줄러의 선입-선처리 알고리즘 프로세스

기존의 공역 상의 경로에 대한 선입-선처리 스케줄링에서는 링크 사이 사이의 노드에 대한 고려를 할 필요가 없었다. 공역 상에서는 공역과 공역 사이의 경계에 대한 제한이 필요하지 않았기 때문이다. 그에 따라

propagation 과정에서 진입하려는 링크의 열려있는 시간 구간과 항공기가 해당 링크에 도달 가능한 시간 구간 사이의 교집합만 수행하였다. 하지만 지상 이동 스케줄링에서는 교차로 제약조건으로 인해 링크와 링크 사이의 노드를 고려해주어야 한다. 따라서 그림 13와 같이 기존 스케줄링에서 구해지는 교집합 결과와 해당 교차로 노드의 진입 가능한 시간 구간에 대해 한 번 더 교집합을 수행한다.

3.4. 링크의 최대 항공기 수용량

지상 이동 스케줄링에서의 링크, 즉 유도로는 길이가 한정되어 있으므로 항공기와 항공기 사이의 안전거리를 확보하기 위해서는 링크가 수용 가능한 항공기의 수에 제한이 있을 수밖에 없다. 링크의 최대 항공기 수용량은 유도로에서의 항공기 분리 간격과 링크 길이에 따라 달라진다. 수용량은 링크의 길이를 분리 간격으로 나눗셈하여 구할 수 있으며, 링크의 길이가 정한 분리 간격보다 작은 경우에는 항공기 최대 수용량을 1로 설정한다. 표3에서 항공기의 정방향과 역방향을 (+)와 (-)로 나타내었는데 이는 링크 내에서 이동 중인 항공기 대수와 관련이 있다. 정방향으로 이동하는 항공기가 링크에 진입하면 링크의 항공기 수는 '+1'이 되며, 반대로 역방향으로 이동하는 항공기가 진입하면 '-1'이 된다.

그림14는 시간에 따른 링크 내 항공기 대수 변화를 나타낸 것이다. 그림14의 좌측 그래프는 정방향으로 이동할 경우의 항공기 진입 가능·불가능 구간을, 우측 그래프는 역방향으로 이동할 경우의 항공기 진입 가능·불가능 구간을 나타낸 것이다. 그림14에서 알 수 있듯이 링크 내에 정방향으로 움직이고 있는 항공기가 존재하는 경우, 역방향으로 이동하는 것이 불가능하며, 그 반대의 경우도 마찬가지이다. 또한 링크에 진입한 항공기가 없는 경우는 방향에 상관없이 항공기 진입이 가능하다.

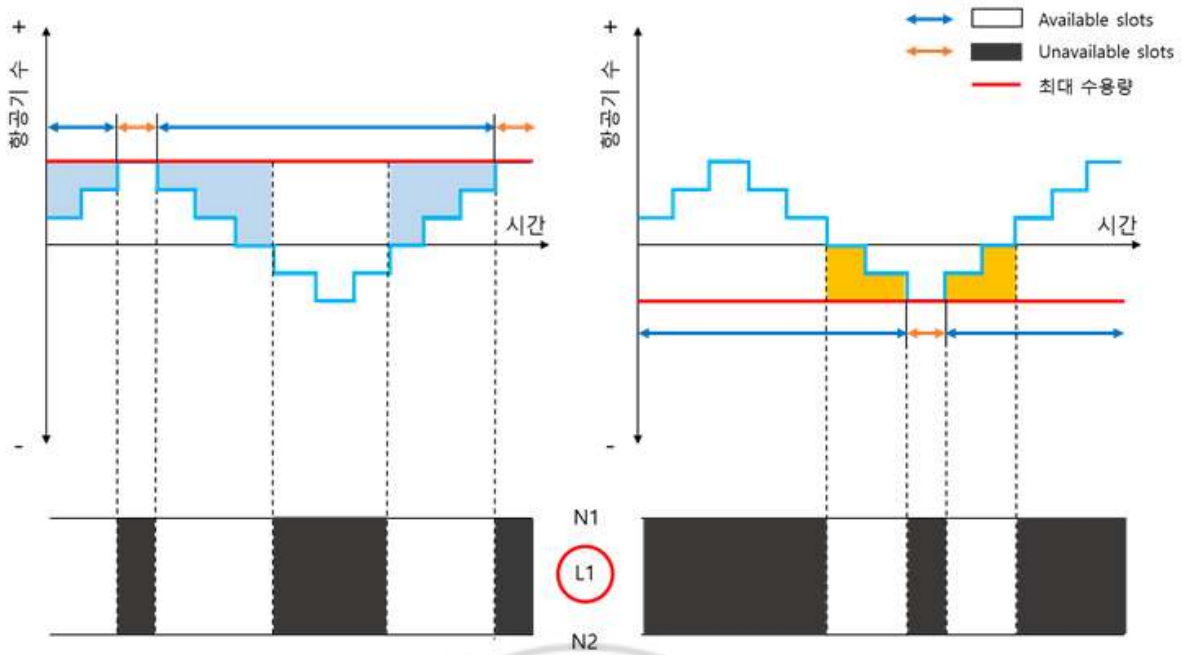

그림 14. 시간에 따른 링크 내 항공기 대수 변화

3.5. 활주로

지상 이동 스케줄링의 경우, 활주로에 대한 제약조건도 고려해야한다. 하나의 활주로 내에서는 한 대의 항공기만이 진행할 수 있다. 따라서 활주로 링크의 최대 항공기 수용량을 1로 지정해준다.

4. 알고리즘 검증

4.1. 링크 최대 수용량 및 링크 방향성 검증

4.1.1. 테스트 모델

개발한 스케줄러의 검증을 위해 임의로 제작한 지상 노드-링크 모델은 그림 15와 같으며, 노드-링크 모델에서 각각의 노드와 링크는 표 4과 같다. 출발하는 항공기의 경로는 N1 또는 N2로부터 N3 - … - N6로 이어지며, 도착 항공기의 경우 N7 - N5 - … - N1/N2로 이어진다.

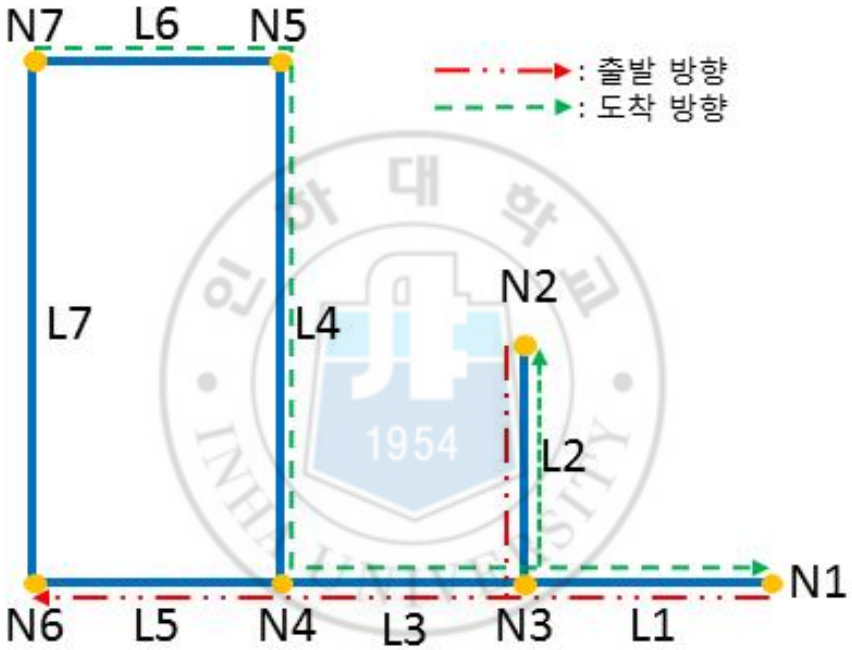

그림 15. 검증용 테스트 노드-링크 모델

표 4. 노드 및 링크 구분

	구 분
N1&N2	게이트 (Gate)
N6	출발 항공기의 이륙 위치
N7	도착 항공기의 착륙 위치
L1-L6	유도로 (Taxiway)
L7	활주로 (Runway)

그림15의 노드-링크 테스트 모델을 기초로 하여 간단한 테스트 시나리오를 만들었다. 시나리오의 총 항공기 대수는 20대이며, 게이트의 turnaround time 및 활주로에서의 항공기 이착륙 간격은 10초로 설정하였다. 이는 개발한 스케줄러의 검증을 위해 항공기간 간격을 실제보다 타이트하게 설정한 것이다. 테스트 시나리오는 표 5와 같다.

표 5. 테스트 시나리오

Flight	Path	Time	D/A
KE1200	N7 → N2	00:00:00	A
OZ8900	N1 → N6	00:00:00	D
OZ8929	N7 → N1	00:00:10	A
LJ302	N2 → N6	00:00:10	D
TW752	N7 → N2	00:00:20	A
KE879	N1 → N6	00:00:20	D
OZ8902	N7 → N1	00:00:30	A
ZE706	N2 → N6	00:00:30	D
TW902	N7 → N2	00:00:40	A
ZE204	N1 → N6	00:00:40	D
BX8100	N7 → N1	00:00:50	A
OZ8002	N2 → N6	00:00:50	D
TW810	N7 → N2	00:01:00	A
LJ304	N1 → N6	00:01:00	D
OZ8906	N7 → N1	00:01:10	A
KE1902	N2 → N6	00:01:10	D
ZE206	N7 → N2	00:01:20	A
LJ562	N1 → N6	00:01:20	D
LJ306	N7 → N1	00:01:30	A
BX8136	N2 → N6	00:01:30	D

표 5의 테스트 시나리오는 출발항공기와 도착항공기 모두 10초 간격으로 출발 및 도착하도록 설정하였다. 앞에서 게이트 turnaround time

과 이착륙 간격을 짧게 준 것과 마찬가지로 스케줄러의 검증을 위해 항공기 스케줄을 조밀하게 구성하였다.

본 테스트에서 각 링크별 항공기의 최대 수용량은 5대로 지정해주었다. 실제 상황에서는 유도로의 길이에 따라 수용 가능한 항공기의 대수가 정해지지만 본 논문에서는 스케줄러 검증을 위해 임의로 5대로 지정한 후 테스트를 진행하였다.

4.1.2. 테스트 결과

표 6. 스케줄링 결과

Flight	Delay (sec)	Time (old)	Time (new)	D/A	Flight	Delay (sec)	Time (old)	Time (new)	D/A
KE1200	0	00:00:00	00:00:00	A	OZ8900	0	00:00:00	00:00:00	D
OZ8929	0	00:00:10	00:00:10	A	LJ302	0	00:00:10	00:00:10	D
TW752	0	00:00:20	00:00:20	A	KE879	0	00:00:20	00:00:20	D
OZ8902	0	00:00:30	00:00:30	A	ZE706	0	00:00:30	00:00:30	D
TW902	0	00:00:40	00:00:40	A	ZE204	265	00:00:40	00:05:05	D
BX8100	105	00:00:50	00:02:35	A	OZ8002	265	00:00:50	00:05:15	D
TW810	225	00:01:00	00:04:45	A	LJ304	375	00:01:00	00:07:15	D
OZ8906	335	00:01:10	00:06:45	A	KE1902	490	00:01:10	00:09:20	D
ZE206	330	00:01:20	00:06:50	A	LJ562	600	00:01:20	00:11:20	D
LJ306	440	00:01:30	00:08:50	A	BX8136	595	00:01:30	00:11:25	D

표6은 테스트 시나리오에 대해 지상 이동 스케줄링을 수행한 결과이다. 표6에서 Time(old)는 기존에 예정되었던 항공기의 출·도착 시간이고, Time(new)는 스케줄링을 통해 새로 계산된 출·도착 시간이다. ‘Delay’항목은 기존의 출·도착 시간에서 새로운 출·도착 시간까지의 지연시간으로, 이 크기만큼 출발 또는 도착을 늦출 경우 공항 내 지상 이동시에 추가 지연 없이 이동이 가능하게 된다. 특히 출발항공기의 경우 출발 시각을 늦춤으로서 유도로에서 무의미하게 소비하는 연료를 줄일

수 있다. ‘D/A’는 출발 항공기인지 도착항공기인지를 구분한 것이다.

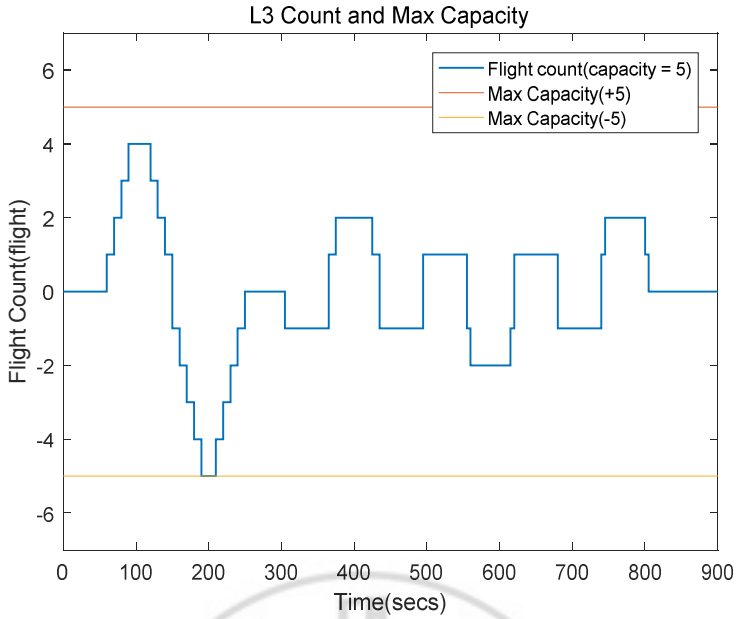

그림 16. 링크 L3의 시간에 따른 항공기 대수 변화(Max.5)

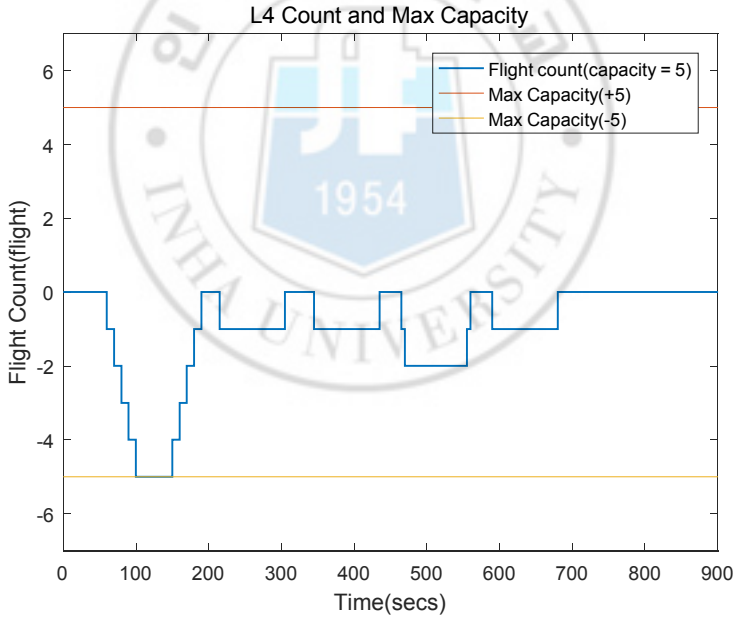

그림 17. 링크 L4의 시간에 따른 항공기 대수 변화(Max.5)

그림16과 그림17은 링크 L3와 L4의 결과를 시간에 따른 링크 내 항공기 대수로 정리하여 그래프로 그린 것이다. 그래프의 x축은 시간, y축은 링크 내 항공기 수를 나타낸다. 항공기 수가 음수로 나타나는 부분은

3.2절에서 언급한 링크의 방향성에 따른 것으로, 항공기가 해당 링크의 음의 방향으로 이동하는 경우이다. 그림16과 그림17에서 모두 링크 내 항공기 대수가 ±5대를 넘지 않는 것을 확인하여, 주어진 제약조건을 만족함을 알 수 있다. 그림18은 제약 조건에 따라 스케줄링 결과가 달라지는 것을 보여주기 위해 링크의 항공기 최대 수용량을 ±3대로 줄이고 테스트한 결과를 첨부한 것이다. 그림16과 그림18의 두 케이스 모두 주어진 제약조건인 최대 항공기 수용량을 넘지 않는 것을 볼 수 있다. 앞의 두 케이스에서 스케줄링을 통해 계산된 전체 항공기에 대한 평균 지연시간은 수용량이 5일 때 3.4분 발생하였으며, 수용량이 3인 경우에는 5.0분으로 지연시간이 늘어난 것을 확인하였다.

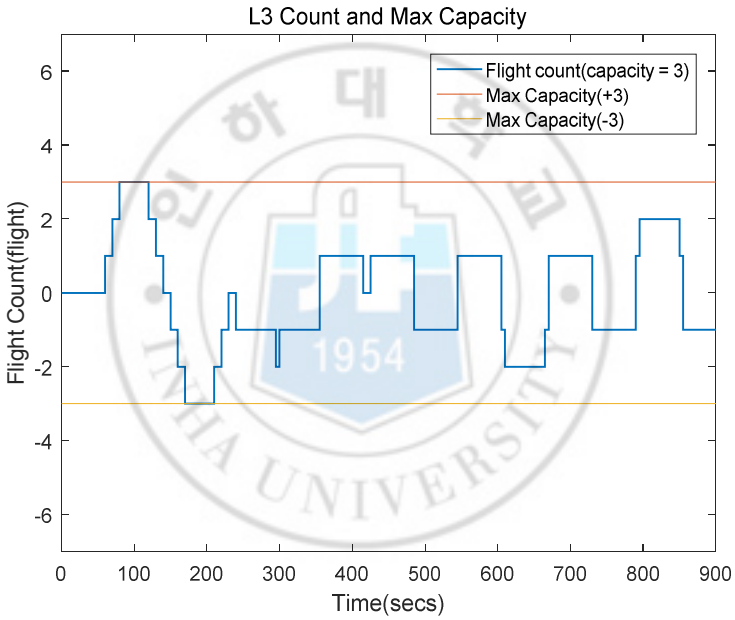

그림 18. 링크 L3의 시간에 따른 항공기 대수 변화(Max.3)

4.2. 교차로 제약조건 검증

4.2.1. 테스트 모델

스케줄러에 적용한 교차로 제약조건을 검증하기 위해 제주공항 노드-링크 모델을 토대로 간단한 시나리오를 작성하였다. 출발항공기(이륙항공기) 225대가 그림 19와 같이 모두 동일한 경로로 이륙위치까지 이동

하며, 제약조건을 확인하기 쉽도록 게이트 출발시간은 모두 0초로 같게 설정하였다. 또한 실제 공항의 게이트는 터미널을 따라 위치해 있지만, 본 테스트에서는 모든 게이트가 유도로 진입 노드까지의 거리가 동일한 위치에 있다고 가정하고 게이트 수는 15개로 설정하였다. 본 테스트는 교차로 제약조건을 확인하기 위한 것이므로, 다른 제약조건에 의한 간섭이 생기지 않도록 링크 최대 수용량을 10대, 활주로 이착륙간격은 1초, 게이트 turnaround time은 30초로 설정하였다.

그림 19. 교차로 제약 조건 검증용 시나리오의 이륙항공기 이동 경로

4.2.2. 테스트 결과

스케줄링 테스트는 교차로 통과시간을 5, 10, 15, 20초로 변경해가며 진행하였다. 표 7은 스케줄링 결과를 정리한 것이다. 표 7의 첫 행은 교차로 통과시간을 노드 제약 조건으로 나타낸 것으로, 기준 시간인 1시간을 통과시간으로 나눈 값이다. 즉, 통과시간이 5초인 경우에는 node rate가 720, 10초일 때는 360, 15초는 240, 20초는 180으로 표현할 수 있다. 교차로 제약조건에 따라 스케줄링 결과가 달라지는 것을 표 7에서 확인할 수 있으며, 교차로 제약 조건을 만족하도록 첫 항공기를 제외한

이후의 항공기들이 통과시간에 따라 지연되는 것을 확인하였다. 또한 노드별 교차로 제약 조건을 다르게 주었을 때는 제약이 더 강한 부분을 만족하도록 스케줄링 되는 것을 확인하였다.

표 7. 교차로 제약조건 테스트 결과

Node rate: 720		Node rate: 360		Node rate: 240		Node rate: 180	
Flight	Delay(sec)	Flight	Delay(sec)	Flight	Delay(sec)	Flight	Delay(sec)
OZ8900	0	OZ8900	0	OZ8900	0	OZ8900	0
KE1200	5	KE1200	10	KE1200	15	KE1200	20
LJ302	10	LJ302	20	LJ302	30	LJ302	40
OZ8928	15	OZ8928	30	OZ8928	45	OZ8928	60
9C8624	20	9C8624	40	9C8624	60	9C8624	80
9C8574	25	9C8574	50	9C8574	75	9C8574	100
KE879	30	KE879	60	KE879	90	KE879	120
TW752	35	TW752	70	TW752	105	TW752	140
ZE706	40	ZE706	80	ZE706	120	ZE706	160
OZ8902	45	OZ8902	90	OZ8902	135	OZ8902	180
…	…	…	…	…	…	…	…

4.3. 이착륙 간격 검증

4.3.1. 테스트 모델

4.2절의 제주공항 모델과 같은 모델에서 제약조건만 변경하여 이착륙 간격에 대해 검증하였다. 4.2절과 동일하게 225대의 항공기가 그림 19의 경로로 이륙하기 위해 이동하는 시나리오에 이착륙 간격 외의 다른 제약조건에 의한 간섭이 없도록 교차로 통과시간은 10초, 게이트 turnaround time은 30초로 설정하였다. 링크의 항공기 최대 수용량은 링크의 길이를 항공기 분리간격으로 나눠준 값으로 설정하였다. 여기서 항공기 분리간격은 Visser와 Roling에 의해 제안된 200m로 설정하였다[7]. 이착륙 간격은 1분과 2분으로 변경하며 테스트를 진행하였다.

4.3.2. 테스트 결과

그림 20과 그림 21는 활주로 진입 바로 전 링크인 AA2의 시간에 따른 항공기 수 변화를 나타낸 것이다. 두 그림 모두 제약조건인 이착륙 간격에 따라 항공기 수가 변하는 것을 확인할 수 있다.

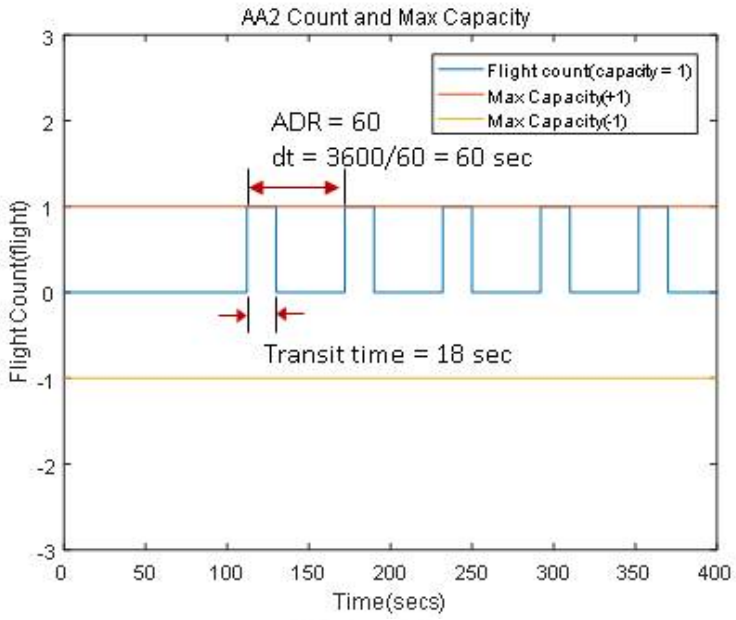

그림 20. 이착륙 간격 1분일 때의 링크 AA2 항공기 수 변화

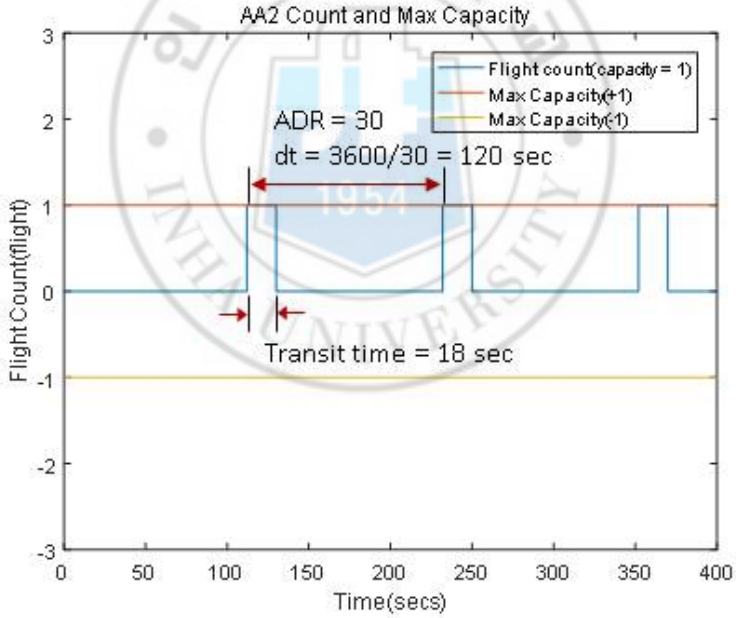

그림 21. 이착륙 간격 2분일 때의 링크 AA2 항공기 수 변화

5. 실 데이터를 이용한 스케줄링 수행 및 결과

지금까지 개발한 항공기 지상 이동 스케줄러에 실제 항공기 이착륙 데이터를 시나리오로 작성하여 스케줄링을 수행하였다. 스케줄링에 이용한 데이터는 제주공항의 FOIS(Flight Operation Information System) 데이터로 3월 31일의 자료를 활용하였다. FOIS 데이터는 각 항공기의 출·도착 게이트와 활주로 정보, 출·도착 시간을 포함하며, 이러한 정보들을 활용하여 시나리오를 제작하였다.

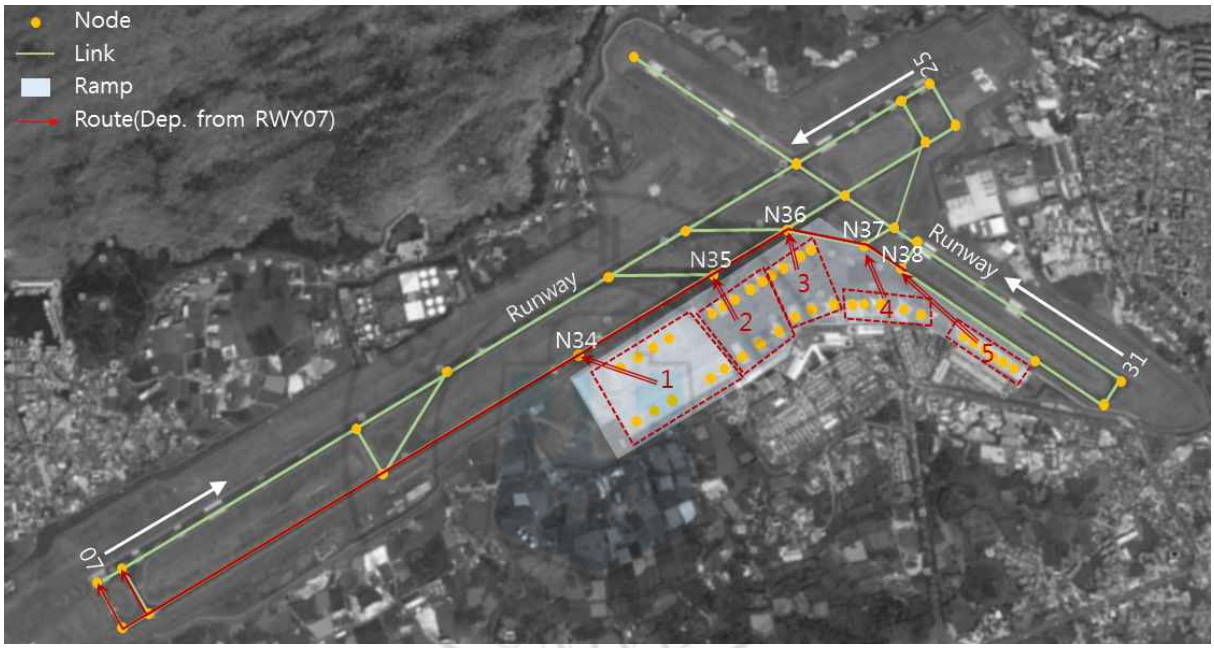

그림 22. 07번 활주로를 이용하는 경우의 출발 항공기 이동경로

항공기의 출발 및 도착 시간은 FOIS 데이터 중 ETD(Estimated Time of Departure)와 ETA(Estimated Time of Arrival)을 기준으로 선정하였으며, 해당 ETD와 ETA는 게이트 기준 출·도착 시간이기 때문에 도착 항공기의 경우 활주로 도착 시간을 경로에 맞춰 재 계산해주었다. 제주공항 노드-링크 모델은 게이트와 유도로, 활주로 정보를 그대로 사용하되, 계류장 내 항공기 이동로의 경우 각 게이트와 유도로 진입점을 직선으로 연결한 링크로 간주하였다. 이때 게이트를 그림 22에서와 같이 크게 5구간으로 나눠 유도로로 진입하는 노드(N34-N37)를 지정하였다. 그

림 22의 화살표는 활주로 07을 이용하는 경우의 출발항공기 이동경로를 나타낸 것이다. 그림 23는 반대로 활주로 25를 이용하는 경우의 출발항공기 이동경로를 나타낸 것이다. 도착항공기의 이동경로는 항공기 기종에 따라 다르게 설정하였는데, 이는 기종별 착륙활주거리가 다르기 때문이다. 착륙활주거리가 짧은 기종의 경우는 그림 23의 C1, B1, B2와 같은 high-speed taxiway를 이용하여 활주로를 빠져나오도록 경로를 선정해주었다.

그림 23. 25번 활주로를 이용하는 경우의 출발 항공기 이동경로

스케줄링에 이용한 총 항공기 수는 412대이며, 공항의 시간별 항공기 수용량을 나타내는 ADR (Aircraft Departure Rate)과 AAR (Aircraft Arrival Rate)은 30, 게이트 turnaround time은 15분으로 설정하였다. 링크 수용량은 링크 길이를 분리 간격으로 나눠 설정했으며, 앞과 동일하게 분리간격은 200m로 지정하였다. 항공기의 이동 속도는 5~20 knot로, 교차로 통과시간은 1~10초로 바꿔가며 스케줄링을 수행하였다.

표8은 항공기 이동 속도와 교차로 통과시간 변화에 따른 결과를 '분' 단위로 정리한 것으로, 항공기 속도 변화에 따라서 그 결과가 많이 달라지는 것을 확인할 수 있다. 개발한 스케줄러를 통해 계산된 결과가 신뢰

성이 있는지를 확인하기 위해 스케줄러로부터 계산된 평균 지연시간을 실제 데이터의 평균 지연 시간과 비교하였다. 실제 평균 지연시간은 FOIS 데이터 중 항공기의 실제 출·도착 시간인 ATD (Actual Time of Departure), ATA (Actual Time of Arrival)와 ETA, ETD의 차 를 평균을 내어 지연시간을 구했으며, 그 수치는 약 12.3분이다. 표8을 살펴보았을 때, 항공기 이동속도가 10-15 knot로 주었을 때의 결과가 실제와 유사한 것을 알 수 있다. 실제 상황에서는 항공기 기종 등에 따라 항공기 지상 활주 속도가 다르다. 하지만 이전 연구들에서 일반적으로 지상 활주 속도를 16 knot로 설정한 것을 토대로 본 스케줄러가 실제와 유사한 결과를 도출할 수 있으며, 실 스케줄링에 사용 가능하다고 판단된다[7][8][9].

표 8. 제약 조건에 따른 평균 지연 (분)

유도로 진입점 통과 시간: 10 sec			
교차로 통과 / 이동속도	10 sec	5 sec	1 sec
5 knots	49.4	49.4	49.4
10 knots	15.4	15.4	14.9
15 knots	11.8	11.8	11.8
20 knots	9.3	9.3	9.3
유도로 진입점 통과 시간: 5 sec			
교차로 통과 / 이동속도	10 sec	5 sec	1 sec
5 knots	48.2	48.5	48.5
10 knots	15.5	17.9	17.9
15 knots	11.7	11.7	11.7
20 knots	8.9	8.9	8.9
유도로 진입점 통과 시간: 1 sec			
교차로 통과 / 이동속도	10 sec	5 sec	1 sec
5 knots	48.2	48.2	47.8
10 knots	15.2	15.8	15.2
15 knots	11.2	11.2	11.2
20 knots	8.9	8.9	8.9

그림 24-27은 표8의 여러 케이스 중 항공기 이동 속도가 15 knot이고, 모든 교차로(유도로 진입점 포함) 통과시간이 5초일 때의 결과 중에서 일부 링크의 시간에 따른 항공기 수 변화 그래프로, 주어진 제약조건인 링크의 최대 수용량을 넘지 않는 것을 알 수 있다.

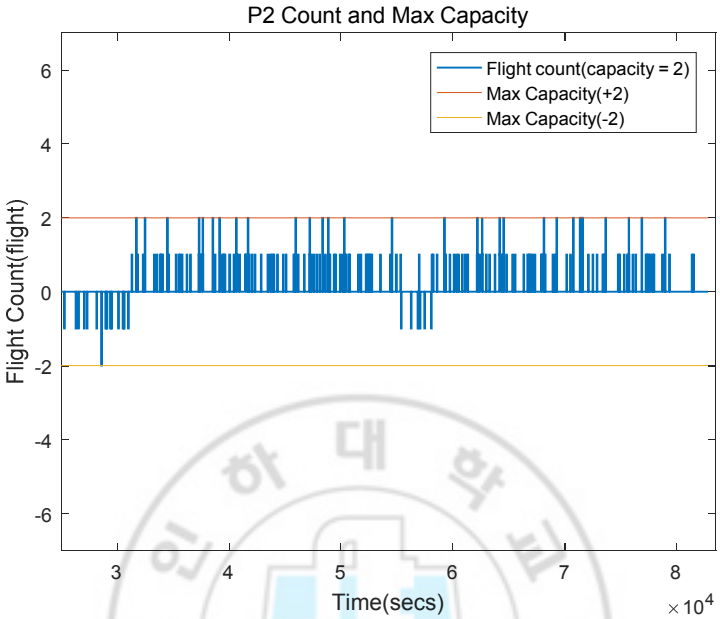

그림 24. 링크 P2의 항공기 수 변화

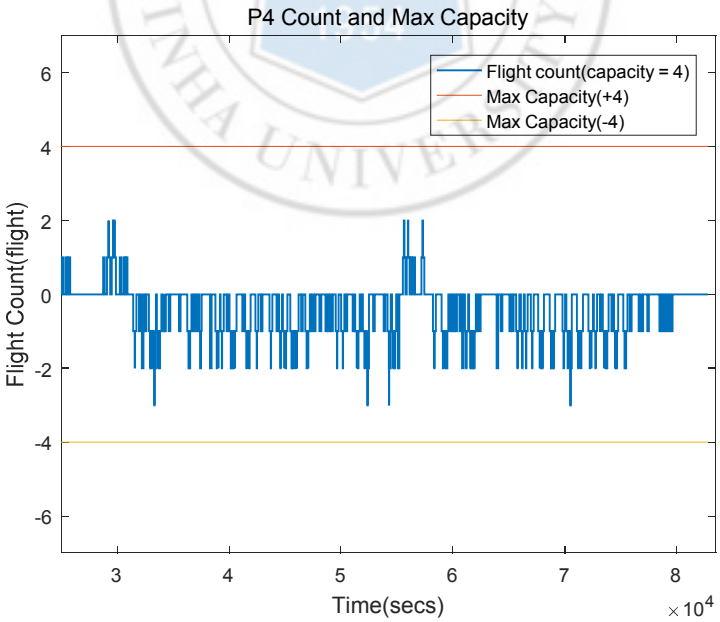

그림 25. 링크 P4의 항공기 수 변화

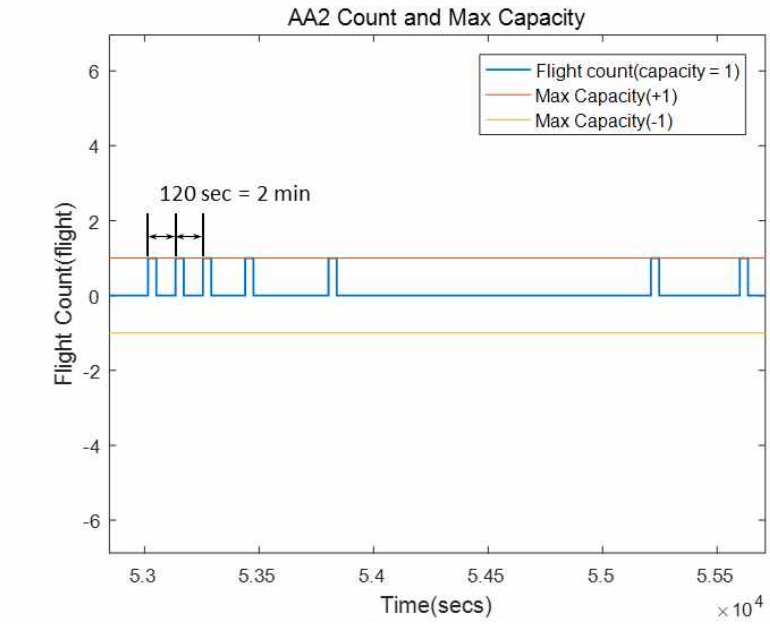

그림 26. 링크 AA2의 항공기 수 변화

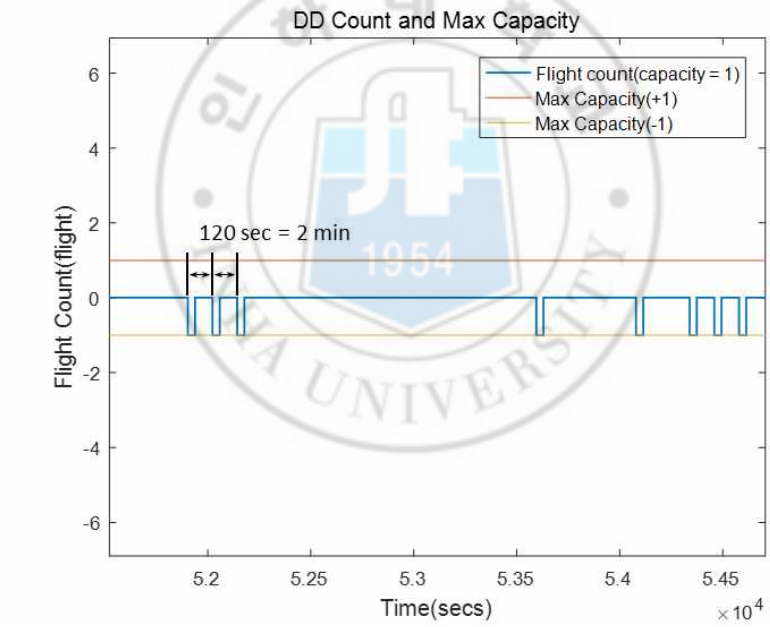

그림 27. 링크 DD의 항공기 수 변화

그림26과 27은 활주로 진입 바로 전 링크의 시간에 따른 항공기 수 변화 그래프로, 시나리오 작성 시에 설정해준 ADR, AAR을 만족하기 위해서는 활주로에서 이착륙 하는 항공기 사이의 시간 간격이 최소 2분이어야 하는데, 이를 만족함을 확인할 수 있다. 그림 28는 동일한 케이스의 지연 시간을 히스토그램으로 나타낸 것이다.

표 9. 속도 변화 허용 시 평균 지연 시간 감소 (분)

속도 변화 / 이동속도	O	X
5 knots	40.3	48.4
10 knots	11.9	17.9
15 knots	10.3	11.7
20 knots	8.7	8.9

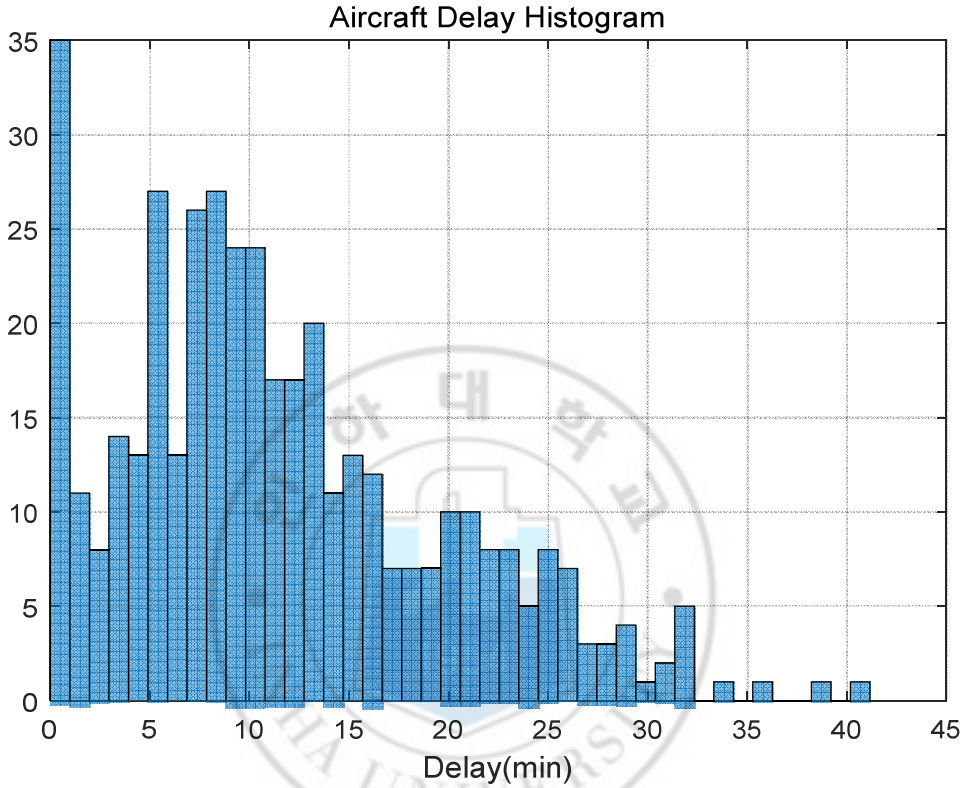

그림 28. 항공기 지연 시간 히스토그램

표 9는 교차로 통과시간 5초일 때의 케이스를 항공기 이동 속도가 항상 일정한 경우와 이동 속도를 최대 5% 증가, 최소 10% 감소 가능한 경우의 두 가지 경우에 대해 스케줄링 한 결과를 비교한 것이다. 항공기 속도 변화가 가능한 경우가 아닌 경우에 비해 평균 지연시간이 확연히 감소하는 것을 확인할 수 있다. 이는 속도 변화에 따라 항공기의 링크 운용 가능 시간 구간이 속도가 일정한 경우에 비해 넓어지기 때문이다. 또한, 실제 상황에서는 회전구간에선 속도를 줄여 운용하며, 타 항공기가 지나가는 것을 기다리기 위해 멈추는 경우도 발생하기 때문에 속도 변화가 있는 경우가 실제와 더 유사하다고 판단된다.

6. 결론

본 논문에서는 선입-선처리 알고리즘을 기반으로 항공기 지상 이동 스케줄러를 개발하였다. 본 스케줄러는 기존에 공역 상에서의 스케줄링을 위해 개발한 스케줄러의 이용 범위를 확대하여 지상 이동 스케줄링에도 활용할 수 있도록 보완한 것이다. 스케줄러의 기본이 되는 선입-선처리 알고리즘에 대해 설명하고, 지상 이동에 있어 고려해야할 제약조건들과 이러한 제약조건을 스케줄링 알고리즘 내에서 어떻게 처리하였는지 설명하였다. 또한 각 제약조건들을 제대로 만족하는지 확인하기 위해 테스트 시나리오를 작성하여 스케줄링을 수행하고, 주어진 조건에 따라 스케줄링 결과가 출력되는 것을 확인하였다. 최종적으로 제주 공항의 실제 항공기 출·도착 데이터인 FOIS 데이터를 활용하여 실제와 동일한 시나리오를 작성하여 스케줄링을 수행하였다. 스케줄링 결과, 스케줄러를 통해 산출한 평균 지연시간과 실제 평균 지연시간의 차이가 미미한 것을 확인하였으며, 이에 따라 본 스케줄러의 결과가 신뢰성이 있다고 판단된다. 또한, 같은 케이스에서 지상 활주 시에 항공기 이동속도 변화가 가능한 경우와 가능하지 않은 경우로 나눠 스케줄링을 수행하여 속도 변화 유무에 따라 그 결과가 변하는 것을 확인하였고, 속도 변화가 있는 경우에 보다 개선된 결과를 제공할 수 있음을 확인하였다.

향후 다양한 일자의 FOIS 데이터를 이용하여 여러 시나리오를 작성하여 반복적으로 스케줄링을 수행 및 결과 분석을 진행할 예정이며, 이를 통해 개발한 스케줄러의 보완을 반복할 예정이다. 또한 도착 항공기에 대한 우선순위 부여와 추가적으로 스케줄러에 추가될 기능에 대해 고려할 예정이다.

7. 참고문헌

[1] 국토교통부 항공정책실, 2016.09, “항공시장동향,” 제 51호, pp. 3~5, 57~61

[2] United States Department of Transportation, http://www.transtats.bts.gov/Data_Elements.aspx?Data=1, 2016.10.19

[3] International Civil Aviation Organization, 2013, *2013-2028 Global Air Navigation Plan*, Doc 9750-AN/963 Forth Edition, Montreal.

[4] Park, C., Lee, H.-T., and Meyn, L. A., 2012, "Computing Flight Departure Times Using an Advanced First-Come First-Served Scheduler," *Proceedings of the 12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, AIAA, pp. 1~8.

[5] Park, B., and Lee, H., “Development of a First-Come First-Served Departure Scheduler,” *7th Asia-Pacific International Symposium on Aerospace Technology (APISAT)*, Cairns, Australia, 25-27 Nov. 2015.

[6] 국토교통부, AIP 9th Edition, ENR 2.1 FIR, UIR, TMA and CTA, 1 May, 2014.

[7] Visser, H. G. and Roling, P. C., “Optimal Airport Surface Traffic Planning Using Mixed Integer Linear Programming,” In *AIAA Aviation Technology, Integration*

and Operations (ATIO) Conference, 2003.

[8] Balakrishnan, H., and Jung, Y., "A Framework for Coordinated Surface Operations Planning at Dallas-Fort Worth International Airport," In *Proceedings of the AIAA guidance, navigation, and control conference*, 2007, pp. 2382-2400.

[9] Malik, W., Gupta, G., and Jung, Y., “Managing Departure Aircraft Release for Efficient Airport Surface Operations,” In *AIAA Guidance, Navigation, and Control Conference*, 2010, pp. 2-5.