사례기반추론과 제약만족문제를 이용한 배차계획시스템 설계

The Design of a Vehicle Fleet Scheduling System using CBR and CSP

전승호*, 김만균**
Seung-Ho Jeon, Man-Kyun Kim

Abstract

In this paper we propose a new framework for vehicle fleet scheduling that integrate a constraint satisfaction problem(CSP) with case-based reasoning(CBR), and this framework is applied to a vehicle fleet scheduling system. Given a user query, CBR module first retrieves similar cases from the case base in which past successful vehicle fleet scheduling cases are stored. Then they are sent to CSP module for making the solutions that simultaneously satisfy all of the constrains and restrictions. Before assigning a feature value, on-line database should be queried to identify whether it is available.

We conducted several examinations to evaluate the performance of our vehicle fleet scheduling system. The result of these examinations show that our framework is more effective to make an optimal solution and is more efficient to save a search time about 8.7% than a methodology that use CSP or CBR independently

Key words : 배차계획시스템, CBR, CSP

1. 서 론

빈번한 교통체증 발생에 따른 차량운행회전율 저하와 인건비 및 유가의 지속적 상승 등은 운송업체의 원가부담을 가중시켜 경영수익 구조를 악화시키는 주요원인이 되고

* 경기공업대학 e-비즈니스과
** 경기공업대학 산업경영시스템과
'03년 6월 접수, 8월 게재확정

있다. 따라서 고객의 만족도와 수익성을 모두 높이기 위해 배차계획을 합리적으로 수립해주는 배차계획시스템의 필요성이 증대되고 있는 실정이다.[1] 배차계획은 다양한 배차관련 제약조건들을 고려하여 언제, 누가, 어떤 차량을 운행할 것인지를 결정하는 것이라 할 수 있다. 지난 40여 년간 Clack과 Wright[2]에 의한 연구를 비롯하여 다양한 배차제약조건을 고려한 무수히 많은 배차계획 관련 연구가 있어 왔음에도 불구하고 국내·외적으로 배차계획시스템의 개발은 다소 미흡한 상태이다. 따라서 실제 현장에서의 배차계획 수립은 이러한 모든 사항을 업무 담당자가 대부분 그동안의 경험에 의한 관례적 또는 임기응변에 의한 수작업의 형태로 이루어지고 있는 실정이다.

본 연구에서는 수요가 동적인 환경에서 실시간 데이터베이스를 이용하여 배차계획을 수립하는 문제를 다룬다. 지금까지 배차계획 문제를 해결하기 위한 방법으로 휴리스틱 및 수리적 접근방법이 많이 연구되어 왔다.[4] 휴리스틱 방법의 장점은 특정 문제에 대해 빠른 가능해를 찾을 수 있지만 다양한 제약조건을 고려하기 힘들다는 단점이 있으며, 수리적 방법도 모델링 자체가 어렵고 탐색시간이 길어질 수 있으며 문제의 성격이 조금만 변해도 다시 모델링 하기 어렵다는 경직성을 가지고 있다. 그 외 인공지능 기법의 하나인 유전자 알고리즘을 이용한 방법과 인공신경망을 이용한 방법 및 제약만족문제를 이용한 방법 등이 연구되었으며 또한 유전자알고리즘과 제약만족문제 또는 인공신경망과 제약만족문제를 복합적으로 이용한 방법 등이 있으나 동적인 환경하에서의 적용에는 한계점을 갖고 있다.[5]

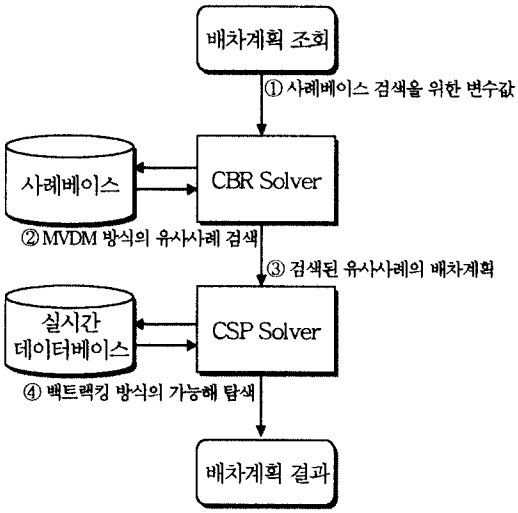
본 연구의 목적은 동적환경에서 일정계획 문제를 해결할 수 있도록 사례기반추론(CBR : Case-Based Reasoning)과 제약만족문제(CSP : Constraint Satisfaction Problem)를 단계적으로 이용하는 새로운 프레임웍을 제시하는데 있다. 구체적으로, 수요가 가변적이며 장비와 인력의 제약을 갖는 문제의 신속하고 정확한 해를 찾기 위해 인공지능 기법 중 하나인 CBR을 이용하여 문제탐색의 범위를 축소하고 실시간 데이터베이스 정보와 제약만족 프로그래밍을 이용하여 도출된 해의 적용 가능성을 검증한다. 이를 위해 광역도시간 버스를 운행하는 여객사의 운행 일정계획 즉, 차량 및 운전자의 배정 및 배차간격을 결정하는 모형을 수립하고 컴퓨터 실험을 통해 제안한 모형의 성능을 평가하고 그 유효성을 검증하였다.

2. 배차계획시스템의 프레임웍

2.1 배차계획시스템의 프로세스

CBR과 CSP를 복합적으로 이용한 배차계획시스템의 새로운 프레임웍은 사례가 저장되는 사례베이스와 유사사례의 검색을 위한 CBR Solver, 제약만족문제의 가능해를 찾

기 위한 CSP Solver 그리고 실시간 데이터를 저장하고 조회하기 위한 데이터베이스로 구성된다. 최적의 배차계획 수립을 위한 배차계획시스템의 프로세스는 <그림 1>과 같다.



<그림 1> 배차계획시스템의 프로세스

먼저 사용자 인터페이스를 통해 배차계획을 조회하면 CBR Solver는 유사도가 높은 사례를 검색하여 검색된 사례에 포함된 변수의 값을 CSP Solver로 보내고 CSP Solver는 온라인 데이터베이스를 조회하여 CBR Solver에서 보내온 변수 값이 이용 가능한지 여부를 확인하여 배차계획 결과값을 출력하게 된다. 만약 CBR Solver에서 보내온 변수 값이 이용 불가능한 경우 온라인 데이터베이스 조회를 통해 이용 가능한 변수 값을 휴리스틱 기법에 의해 차례로 적용한 후 최종 배차계획의 가능해를 출력한다.

2.2 배차계획시스템 구성 모듈

(1) 사례베이스

사례베이스는 과거의 사례들로 구성된다. 사용자의 조회내용과 그에 대응하는 일정계획은 하나의 쌍을 이룬다. 그러므로 사례베이스는 같거나 매우 유사한 조회들과 일정계획의 반복적인 사례들을 저장 또는 삭제할 수 있다.

(2) CBR Solver

사례의 검색은 과거 사례와 현재의 조회내용간의 유사도에 따라 사례베이스로부터 유사한 사례들이 검색된다. 유사한 사례들이 현재의 상황에서 이용 가능한지를 확인해야 하기 때문에 직접적인 일정계획으로 사용되지 못하며 CSP Solver에 의해 검증단계를 거치게 된다.

(3) CSP Solver

CBR Solver를 통해 검색된 사례를 구성하는 변수 값들이 제약변수로서 이용 가능한 상태인지를 온라인 데이터베이스를 통해 확인한 후 가능해 값을 찾는다. 만약 검색된 사례를 구성하는 값들이 이용 불가능한 상태인 경우 휴리스틱 기법을 이용하여 적용가능한 제약변수 값을 대입하여 가능해를 찾게 된다.

(4) 온라인 데이터베이스

운전자 관련 데이터, 버스 관련 데이터,

버스의 운행과 관련된 데이터 등 SP Solver가 가능해를 찾기 위해 고려하는 제약식의 모든 변수를 포함한다.

2.3 유사도 측정방식

사용자의 조회내용과 유사한 사례는 사례베이스로부터 도출된다. 사용자의 조회내용을 속성값 a_j를 갖는 Q, 속성값 b_j를 갖는 사례베이스에 저장된 하나의 사례를 C라 하고 조회내용 및 사례가 갖는 속성의 개수를 m이라 하면 식 (1)과 같다

$$Q=[a_1, a_2, ..., a_m],\ C=[b_1, b_2, ..., b_m] \quad (1)$$

조회내용과 사례간의 차이는 식 (2)와 같이 각각의 속성값들의 차이를 합한 것과 같다.

$$d(Q, C) = \sum_{j=1}^{m} \delta(a_j, b_j) \quad (2)$$

($\delta(a_j, b_j)$는 조회내용 및 사례의 j번째 속성간의 차이임)

사례베이스로부터 하나 또는 일련의 유사사례를 검색하기 위해 일반적으로 사용되는 두 개의 검색기술은 Nearest Neighbor Retrieval과 Inductive Retrieval이 있다.[3] 본 연구에서는 사용상의 편리성을 제공하는 Nearest Neighbor Retrieval을 선택한다. 유사한 사례를 찾기 위해 사례들 간의 유사도를 계산해야 한다. 유사도의 측정은 MVDM (Modified Value Difference Metric) 기법을 이용한다.

MVDM 기법은 카테고리에 포함되는 명목 값을 갖는 변수로 구성된 사례의 각 변수간의 거리를 측정하는데 가장 적합하다고 여겨지는 새로운 기법이다.[6] 이 기법을 이용하여 학습을 위한 사례의 집합을 토대로 모든 변수 값들 간의 거리를 정의하는 하나의 매트릭스가 통계적으로 도출된다. 특정 변수값에 대한 두 값의 거리 δ는 식 (3)과 같이 정의된다.

$$\delta(V1, V2) = \sum_{i=1}^{n} \left| \frac{C1_i}{C1} - \frac{C2_i}{C2} \right|^k \quad (3)$$

위 식에서 $V1$과 $V2$는 특성을 나타내는 두 개의 가능한 값들이다. 두 값들 간의 거리는 모든 n개의 클래스들에 대한 합이다. $C1_i$는 $V1$이 범주 i로 분류된 횟수이며, $C1$은 값 $V1$이 발생한 총 횟수이다. k는 상수이며 일반적으로 1 값을 갖는다.

우리는 조회내용과 사례베이스에 저장된 사례간의 차이가 임계치 값 ε 보다 작은 사례들을 유사사례로 검색될 수 있도록 정의한다. 즉 $d(Q, C) \leq \epsilon$ 인 경우라면 C는 Q에 대한 유사사례로 검색된다. 여기서 ε 값은 최소 3개의 사례가 사례베이스로부터 검색될 때까지 증가할 수 있도록 조정하여 유사사례가 검색되지 않는 경우를 방지토록 한다.

2.4 CSP 탐색방식

제약만족문제 기법은 비교적 최근에 연구되고 있는 분야이며 경영과학과 인공지능을 결합한 형태라고 할 수 있다. 제약만족문제는 제한된 영역의 범위를 갖는 변수들에 주어진 제약조건들을 모두 만족하는 값을 할당하는 문제를 말한다. 특히 이러한 제약만족문제라 부르는 문제 유형을 푸는 것을 목적으로 하는 프로그래밍을 제약 프로그래밍이라 하는데 CSP Solver는 일종의 제약 프로그래밍 기능을 수행하게 된다.

제약 프로그래밍은 기본적으로 모든 변수가 가질 수 있는 값들의 조합을 나열하여 가능해나 최적해를 찾아 나간다. 그러나 이 모든 조합 하나하나를 모두 따져가며 탐색하기에는 시간이 너무 오래 걸릴 수 있으므로 이 기법의 핵심은 어떻게 하면 효율적으로 탐색공간을 줄여서 해를 찾는데 까지 소요되는 시간을 줄일 수 있는가에 있다. 탐색 공간을 줄이는 검색기법으로는 constrain propagation, arc consistency, backtracking, forward checking 등이 있다. CSP Solver는 CSP를 적용한 문제에서 지금까지 일반적으로 가장 많이 쓰이는 휴리스틱 탐색기법인 backtracking 방식을 사용한다.

Backtracking 방식은 변수의 선택에 있어 도메인이 가장 작은 변수를 우선적으로 선택하여 탐색 실패의 가능성을 줄여 그만큼 탐색횟수를 줄일 수 있지만 CSP Solver는 변수를 할당하는 단계에서 CBR Solver에서 검색된 유사사례의 배차계획 결과값을 우선적으로 선택한다. 그렇게 하는 이유는 유사사례로 검색된 사례의 결과가 배차업무담당자의 노하우를 포함하고 있으며 또한 일정한 배차계획의 패턴을 갖고 있을 가능성이 있기 때문에 이를 반영하기 위함이다. 이 때 CBR Solver에서 검색된 유사사례들의 각각의 배차계획 관련 결과값들이 현재 이용 가능한 상태인지를 실시간 데이터베이스에서 조회하여 이용여부를 결정하는데, 만일 이용 불가능한 상태라면 그 변수가 속한 속성값 도메인에서 이용 가능한 값을 취한 후 변수에 할당한다. 그 후 그 변수가 걸려있는 모든 제약조건에 그 변수에 할당된 값을 넣고 모든 변수가 어떤 값을 취할 때가지 계속 이 절차를 실행한다. 만약 실패를 하게 되면 backtracking을 한다.

CSP Solver는 변수들의 선언과 그 변수들에 관련된 제약조건 집합만 선언해 주면 되기 때문에 프로그램이 간결하고 모듈화할 수 있다는 장점이 있어 문제의 내용이 바뀌어도 쉽게 모델링을 수정할 수 있다는 장점이 있다.

3. 문제의 정의

배차계획 문제는 적용되는 상황에 따라 문제의 성격이나 크기가 달라진다. 그러나 일반적인 배차계획 문제의 공통점은 고객의 수요를 충족시켜야 하고 작업장 내의 여러

가지 제약조건들을 만족시켜야 한다는 점이다.

본 연구에서 제안한 배차계획시스템은 광역도시간 노선을 운행하는 K여객의 배차계획 업무에 적용하기 위해 설계되었다. K여객의 버스운행 시간은 오전 5 : 00부터 새벽 1 : 00까지(25시간)이며 운행노선을 1회 왕복하는데 소요되는 시간은 평균 2시간, 배차간격은 평균 15분이다. 배차계획을 수립할 때에는 교통상황에 따라 탄력적인 배차간격을 유지함으로서 승객의 대기시간을 최소화하고 운전자의 연속근무시간의 상한 및 적절한 휴식시간의 제공을 통한 사고의 위험을 줄이며 일정수준의 탑승률 유지를 통한 경영의 합리화를 달성할 수 있는 방안이 고려되어야 한다.

3.1 제약조건

배차계획의 주요 내용은 시간대별 배차간격, 운전자 및 운행차량을 결정하는 것이다. CSP Solver는 운송회사의 정책과 승객의 요구 및 운전자의 편의를 모두 고려하여 경영수익의 개선, 고객에 대한 서비스 향상 및 운전자의 이직률 감소 등과 같은 목적함수를 극대화 할 수 있는 결과를 도출하기 위해 다양한 필수제약조건과 선택제약조건을 갖을 수 있다. 필수제약조건은 반드시 지켜져야 하고 모든 운전자에게 평등하게 적용되며, 선택제약조건은 필수제약조건이 모두 만족되는 가능해들 가운데 가능하면 더 만족되었으면 하는 제약조건으로 가능해가 나오는 조건에서 더 강화 또는 완화될 수 있다. 아래의 제약조건 (1)~(5)는 필수제약조건으로, (6)~(7)는 선택제약조건으로 구분한다.

(1) 작업수요

각 시간대별 요구되는 버스의 수요는 확률적이다. 따라서 출 · 퇴근 시간과 같이 승객수요가 증가하는 시간대는 배차간격을 줄여주어야 하며 사고와 같은 돌발사태가 발생하였을 때 특별차량을 추가 투입하는 등 불확실한 수요에 대한 유연한 대처방안을 고려해야한다.

(2) 총 근무시간

각 운전자들은 모든 계획시간 동안 근무할 수는 없다. 각 운전자의 총 근무시간 한계 내에서만 근무가 가능하다.

(3) 운전자의 운행버스

한 운전자가 동일 시간대에 2대 이상의 버스에 탑승할 수 없다.

(4) 연속근무시간과 휴식

한 운전자가 너무 오랫동안 근무할 경우 피로도가 쌓여 안전운전에 지장을 초래할 수 있다. 따라서 연속 근무시간 후에는 얼마간의 휴식이 필요하다.

(5) 1회 최소 연속탑승시간

운전자는 최소 왕복코스 1회 이상을 운행

해야 한다. 즉, 운행 중 운전자가 바뀔 수 없다.

(6) 연속휴무선호

운전자가 분산된 휴식시간 보다는 연속된 휴식시간을 선호하는 경향이 강하다. 따라서 가능하면 휴식시간이 연속되게 계획될 수 있도록 한다.

(7) 각 운전자의 특별한 요구

운전자가 특정한 시간에 근무할 수 없는 경우가 생기거나 휴식을 요청할 경우 또는 근무를 요청할 경우 가능한 범위 안에서 최대한 반영하도록 한다.

3.2 제약조건의 표현

비록 CSP Solver가 가능해를 찾기 위해 고려하는 제약조건은 총 7가지지만 이들 모두를 식으로 표현할 필요는 없다. 위의 필수제약조건 중 한 운전자가 동일 시간대에 2대 이상의 버스에 탑승할 수 없다는 필수제약조건과 운행 중 운전자가 바뀔 수 없다는 필수제약조건 및 근무자가 선호하는 시간대에 운행할 수 있도록 하는 선택제약조건은 실시간 데이터베이스의 내용을 조회하여 반영할 수 있기 때문이다. 따라서 제약식을 필요로 하는 제약조건들에 대한 표현은 다음과 같다.

(1) 요구되는 시간대별 버스운행의 수요

$$\sum_{j=1}^{m} x_{ij} \geq d_i \quad (i=1, 2, \ldots, s, j=1, 2, \ldots, m)$$

x_{ij} 는 시간대 i에서의 버스 j에 탑승하는 운전자 번호를 도메인으로 하는 의사결정변수, m은 전체 버스의 수, s는 계획시간 동안의 전체 시간대 수, d_i는 시간대 i 에서의 일정계획상 작업에 투입되는 버스의 대수이다.

(2) 운전자의 총 근무시간

$$\sum_{i=1}^{s} Y_{ijk} \leq max \ (j=1, 2, \ldots, m, \ k=1, 2, \ldots, n)$$

Y_{ijk} 는 운전자 k가 i 시간대에 j 버스에 탑승하는 경우, n은 운전 가능한 운전자의 수, max는 한 운전자가 한 기간동안 버스에 탑승할 수 있는 최대시간의 한계이다.

(3) 운전자의 연속근무시간과 휴식

$$Y_{ijk} + Y_{(i+1)jk} + \cdots + Y_{(i+q)jk} \leq r$$

$(i= 1, 2, \ldots, s-q, \ k= 1, 2, \ldots, n, \ j= 1, 2, \ldots, m)$ (단, $1<q<s-2$, q, r은 정수)

3.3 변수의 정의

(1) 입력변수

CBR Solver가 유사도를 측정해서 유사도가 높은 과거의 사례를 검색하기 위해 사용자가 입력해야할 할 변수는 모두 6개 이며 이들은 <표 1>과 같이 5개의 범주로 구분된 영역 중 하나로 분류되어 MVDM 기법을 이용한 유사도 측정에 사용 된다.

<표 1> 입력변수와 범주의 정의

속성 \ 범주	1	2	3	4	5
시 간 대	05 : 00 ~ 06 : 29	06 : 30 ~ 09 : 29	09 : 30 ~ 17 : 29	17 : 30 ~ 19 : 29	19 : 30 ~ 24 : 00
계 절	1월 ~ 2월	3월 ~ 6월	7월 ~ 8월	9월 ~ 11월	12월
요 일	일 (휴일)	토	월	화,수, 목	금
운행시간 (회차지점)	50분 미만	50분 ~ 59분	60분 ~ 69분	70분 ~ 79분	80분 이상
운행시간 (출발지점)	50분 미만	50분 ~ 59분	60분 ~ 69분	70분 ~ 79분	80분 이상
탑 승 률	10명 미만	10명 ~ 19명	20명 ~ 29명	30명 ~ 40명	40명 이상

범주의 분류 기준은 승객 수요에 있어 유사한 패턴을 보이는 계절, 요일, 시간대를 기준으로 하였으며, 운행시간은 평균 1시간이 소요되므로 출발지점에서 회차지점까지의 소요시간 및 회차지점으로부터 출발지점까지의 소요시간을 모두 60분 기준으로 분류하였다.

(2) 출력변수

배차계획 문제는 결국 언제 어떤 버스를 어떤 운전자가 운행해야 하는가를 결정하는 것이므로 <표 2>와 같은 매트릭스의 빈칸에 운전자를 할당하는 것을 의미한다. 따라서 배차계획시스템의 출력변수는 {출발시각, 운전자, 버스}의 집합으로 표현할 수 있다.

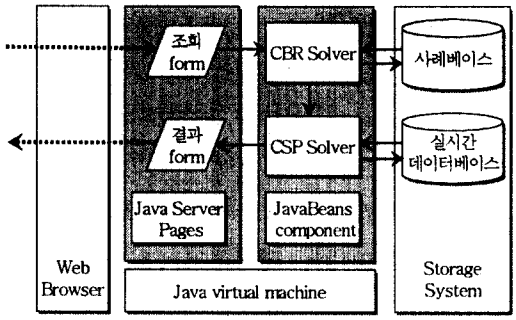
<표 2> 출력변수의 정의

시간 \ 버스	버스 1	버스 2	버스 3	·····	버스 n
출발시각 1	운전자 A				
출발시각 2		운전자 B			
출발시각 3			운전자 C		
⋮				·····	
출발시각 m					운전자 J

4. 시스템의 설계

4.1 배차계획시스템의 기술구조

배차계획시스템의 구조는 <그림 2>와 같다. 이 시스템은 JSP와 Java2 기술로 실행된다. JSP는 동적인 웹 페이지를 생산하고 사용자는 JSP에 의해 생성되는 HTML 문서상의 조회 form에 관련 데이터를 입력하게 된다. 사용자의 조회문서는 JavaBeans 컴포넌트 형태로 수행되는 CBR Solver로 전송된다. CBR Solver는 검색된 유사사례들을 구성하는 각각의 속성값들이 현재 이용가능한 상태인지를 확인하기 위해 실시간 데이터베이스를 조회한 후 가능해를 구하고 그 결과를 결과 form에 출력한다.



<그림 2> 배차계획시스템의 기술구조

4.2 데이터베이스 구조

온라인 데이터베이스의 구조는 <표 3>과 같이 3개의 테이블을 갖는다. 운행 테이블은 현재 운행중인 운전자와 운전자가 탑승한 버스 및 출발시간, 도착시간 등의 운행에 관련된 정보를 저장하고 있어 CBR Solver로부터 도출된 유사사례에 포함된 배차계획 변수의 속성값들이 현재 이용 가능한 상태인지를 파악할 수 있게 한다.

<표 3> 데이터베이스의 구조

Table	Field	형식	설 명
운전자	운전자 ID	일련번호	
	운전자 명	텍스트	
	근무 상태	텍스트	근무가능 상태와 불가능 상태로 구분
버스	버스 ID	일련번호	
	차량 상태	텍스트	운행 가능 상태와 불가능 상태로 구분
운행	운전자 ID	일련번호	
	버스 ID	일련번호	
	출발시간	날짜/시간	출발지점에서 버스
	소요시간 1	정수	출발지점에서 회차지점까지 소요된 시간
	소요시간 2	정수	회차지점에서 출발지점까지 소요된 시간
	총 승객 수	정수	출발지점부터 종점까지 탑승한 총 승객 수

4.3 사례베이스 구조

K여객의 실제 배차일정계획 자료를 바탕으로 <표 4>와 같이 사례베이스를 구축하고 배차계획 결과를 도출하였다.

<표 4>사례베이스의 구조

속성 / 사례	시간대	계절	요일	소요시간1	소요시간2	탑승률
:	:	:	:	:	:	:
00345	2	3	3	4	3	5
00346	2	3	4	3	3	4
00347	2	3	4	4	3	3
:	:	:	:	:	:	:

5. 실행결과 및 성능평가

5.1 실행결과

CSP Solver를 이용하여 매 시간대별 배차계획을 도출하기 위해 고려된 총 운전자 수는 21명, 버스의 총 대수는 16대 이다. 배차계획시스템을 이용하여 각 시간대별 배차

계획의 가능해를 출력하여 버스와 시간대의 매트릭스 형태인 <표 5>의 결과를 얻었다.

<표 5>에서 버스와 시간대로 구성된 배차계획 매트릭스 내에 있는 알파벳은 운전자를 뒤의 숫자는 같은 시간대 내에서의 운행순서를 의미한다. CSP Solver가 도출한 가능해는 시간대별 승객의 수요를 고려하여 배차량을 탄력적으로 운영하도록 되어 있으며 운전자가 선호하는 근무시간을 고려한 운행일정 및 충분한 휴식시간을 줄 수 있도록 모든 제약조건을 만족하는 배차계획이 도출되었음을 알 수 있다

5.2 성능 평가

CBR Solver의 성능을 평가하기 위해 시간대별 승객수요 예측능력을 기준으로 평가하였다. K여객의 운행데이터를 이용하여 일부는 사례베이스에 저장하고 그 이후에 발생된 50건의 다른 사례들을 이용하여 <표 6>과 같은 평가 결과를 얻었다.

<표 6>의 결과에서 사례베이스의 크기가 커질수록 적중률이 증가함을 알 수 있었다. 그러나 실험 4~7의 결과는 76%~77% 사이에서 큰 차이를 보이지 않고 있어 사례베

<표 5> 배차계획 매트릭스

버스 / 시간대	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
05 : 00~06 : 00	P/4	B/1	C/2	D/3												
06 : 00~07 : 00					E/1	F/2		H/3								T/4
07 : 00~08 : 00		B/5	C/6	D/7							K/1	L/2	M/3		O/4	
08 : 00~09 : 00					**E/4**	**F/5**	**G/1**	**H/6**	**R/3**					**U/2**		
09 : 00~10 : 00										J/1	K/2	L/3	M/4		O/5	
10 : 00~11 : 00	A/1	B/2	C/3	D/4												
11 : 00~12 : 00					E/2	F/3	G/1	H/4								
12 : 00~13 : 00										J/1		L/2	M/3		O/4	
13 : 00~14 : 00	A/2	B/3		D/4										N/1		
14 : 00~15 : 00			C/1			F/3	G/2	H/4								
15 : 00~16 : 00									I/2			U/3	M/1	N/4		
16 : 00~17 : 00		P/4	Q/1	R/2	S/3											
17 : 00~18 : 00						T/1				J/3	K/2				O/4	E/5
18 : 00~19 : 00	A/1		Q/6						I/3			L/2	M/4	N/5		
19 : 00~20 : 00		**P/1**		**R/4**	**S/5**	**T/6**	**G/2**	**U/3**			**K/7**					
20 : 00~21 : 00									I/5	J/1		L/4			O/2	E/3
21 : 00~22 : 00	A/2		Q/1		S/4									N/3		
22 : 00~23 : 00				R/2		T/3		U/1		J/4						
23 : 00~24 : 00		P/1	Q/3		S/4				I/2							

이스의 수가 늘어감에 따라 적중률이 비례하여 늘지 않고 있음을 알 수 있다. 결과적으로 CBR Solver의 승객 수요예측 능력은 77%의 정확성이 있다고 말할 수 있으며, 예측능력을 향상시키기 위해 다양한 유사도 측정 기법을 적용하여 실험할 수도 있으나 우리나라의 불규칙적인 교통환경을 감안했을 때 이는 비교적 높은 예측력이라 평가할 수 있다.

<표 6> CBR Solver의 수요 예측력

실험 횟수	사례베이스의 크기(개)	실험한 사례 수(개)	적중률(%)
1	100	50	59.0
2	150	50	67.3
3	200	50	73.3
4	250	50	75.3
5	300	50	76.0
6	350	50	77.3
7	400	50	76.6

시간대별 배차계획을 도출하기 위해 고려된 총 운전자 수는 21명이고 버스의 총 대수는 16대이므로 이것은 $22^{(19*16)}$개의 탐색범위를 갖는 대단히 큰 문제라 할 수 있다. 본 연구에서 설계한 배차계획시스템을 펜티엄(IV), 256MB의 컴퓨터에서 실행하였을 때 시간대별로 가능해를 찾는데 소요된 시간은 평균 21초였다. 한편 CBR Solver에서 도출된 배차계획 결과를 활용하지 않고 온라인 데이터베이스 만을 이용했을 경우 시간대별로 가능해를 찾는 데 소요된 시간은 평균 24초였다. 따라서 CBR을 복합적으로 이용했을 때 약 8.7%의 성능이 향상되는 효과가 있음이 검증되었다.

6. 결 론

본 연구에서는 수요와 자원이 가변적인 동적환경 하에서 배차계획을 수립하는 문제를 해결할 수 있도록 CBR과 CSP 및 온라인 데이터베이스가 순차적으로 이용된 새로운 프레임웍을 제시하였으며 새로운 프레임웍을 적용한 배차계획시스템을 설계하고 광역도시간 버스운송사의 배차계획 문제에 적용하여 그 유효성을 평가하였다.

유사도 측정기법 중 하나인 MVDM을 적용한 CBR Solver는 승객 수요예측 적중률을 평가하기 위해 50개의 테스트 자료를 시험한 결과 약 77%의 적중률을 보였으며, CBR Solver의 결과를 CSP Solver에 이용했을 경우가 CSP Solver를 단독으로 사용했을 경우에 비해 탐색속도에 있어서도 약 8.7%의 성능향상이 되고 있음을 보여 CBR과 CSP를 복합적으로 이용한 방식이 효과가 있음을 보였다.

본 연구에서 제안한 배차계획시스템의 장점은 첫째, 실시간 데이터베이스를 활용하여 동적인 환경에서도 모델의 변경 없이 배차계획을 수립할 수 있다는 점이다. 단순히

제약조건의 파라메터를 변경하여 보다 유연하게 배차일정 정책을 반영한 계획이 수립할 수 있다. 둘째, CBR Solver가 제공한 기본해에는 배차간격, 운전자의 선호시간대 및 선호차량, 적절한 휴식기간의 길이 등 오랜 경험에서 터득한 배차계획담당자의 노하우가 반영되어 있어 CSP가 고려해야 하는 제약조건을 만족하는 기본해 뿐만 아니라 최적해에 근접한 배차계획을 수립하는데 기여할 수 있다는 점이다. 셋째, 신속한 배차계획을 수립할 수 있으며 향후 정보기술의 적용으로 데이터 수집이 용이하게 되면 배차계획 의사결정을 자동화 할 수 있어 인력감축의 효과를 얻을 수 있다는 점이다.

본 연구에서 설계한 배차계획시스템의 사용으로 정확한 배차관리가 이루어지면 승객 대기시간의 감축 및 탑승률 제고, 배차인력 절감 및 운전자의 이직률 감소 등을 통한 경영합리화로 수익이 극대화 될 것으로 기대된다. 또한 동적인 환경에서 일정계획을 수립해야 하는 산업분야 및 공공분야 에서도 본 연구에서 제안한 새로운 프레임웍은 효과적으로 적용될 수 있을 것으로 기대된다.

참고문헌

[1] 양병희, "다목적 최적화를 고려한 배차계획시스템", 한국경영과학회지, V.19, No. 3, pp. 63~79, 1994

[2] Clarke, G. and J. W. Wright, "Scheduling of Vehicles from a Central Depot to a Number of Delivery Points", Operations Research, Vol. 12, pp. 568~581, 1964

[3] Francisco, R. and P. Avessani, "Learning a Local Similarity Metric for Case-Based Reasoning," Procedings in International Conference on Case-Based Reasoning, pp. 301~312, 1995

[4] Sqalli, M and Freuder, E. C., "Integration of CSP and CBR to Compensate for Incompleteness and Incorrectness of Models", AAAI Spring Symposium on Multimodal Reasoning, Stanford University, pp. 74~79, 1998

[5] Sariklis, D. and S. Powell "A heuristic method for the open vehicle routing problem", Journal of the Operation Research Society, Vol. 51, pp. 564~573, 2000

[6] Wilson, D. R. and T. R. Martinez, "Improved Heterogeneous Distance Function," Journal of Artificial Intelligence Research, Vol. 6, pp. 1~34, 1997