한국경영과학회지
제41권 제1호
2016년 2월
http://dx.doi.org/10.7737/JKORMS.2016.41.1.041

공공 자전거 정적 재배치에의 VNS 알고리즘 적용

임 동 순†
한남대학교 산업경영공학과

Application of Variable Neighborhood Search Algorithms to a Static Repositioning Problem in Public Bike-Sharing Systems

Dong-Soon Yim
Department of Industrial and Management Engineering, Hannam University

■ Abstract ■

Static repositioning is a well-known and commonly used strategy to maximize customer satisfaction in public bike-sharing systems. Repositioning is performed by trucks at night when no customers are in the system. In models that represent the static repositioning problem, the decision variables are truck routes and the number of bikes to pick up and deliver at each rental station. To simplify the problem, the decision on the number of bikes to pick up and deliver is implicitly included in the truck routes. Two relocation-based local search algorithms (1-relocate and 2-relocate) with the best-accept strategy are incorporated into a variable neighborhood search (VNS) to obtain high-quality solutions for the problem. The performances of the VNS algorithm with the effect of local search algorithms and shaking strength are evaluated with data on Tashu public bike-sharing system operating in Daejeon, Korea. Experiments show that VNS based on the sequential execution of two local search algorithms generates good, reliable solutions.

Keywords : Public Bike System, Static Repositioning, Local Search Algorithm

논문접수일 : 2015년 07월 01일 논문게재확정일 : 2015년 11월 11일
논문수정일(1차 : 2015년 08월 20일, 2차 : 2015년 10월 13일)
† 교신저자, dsyim@hnu.kr

1. 서 론

공공 자전거 대여 시스템은 대여소에서 자전거를 빌려 단기간 사용한 후 원하는 곳에 반납할 수 있는 서비스를 제공하는 것으로 2014년 8월 전 세계 600개 이상의 도시에서 운영되고 있다. 대여 시스템에서 이용자에 대한 서비스 질을 높이기 위해서는 이용자가 원하는 어느 때에나 이용이 가능하여야 하고, 원하는 장소에서 반납이 가능하여야 한다. 그러나, 자전거 수가 한정되어 있어 이용자가 원하는 시간과 장소에서 이용할 수 없는 상황이 발생한다. 또한, 무인 대여소에서 자전거를 보관하기 위한 거치대의 수가 한정 되어 있어 더 이상의 자전거 반납이 불가능한 경우가 발생한다. 이러한 서비스 불만족을 해결하기 위한 문제는 공공 자전거 대여 시스템뿐만 아니라 렌트카를 포함하는 여러 수송 시스템에서 발생한다. 이러한 시스템에서 고려되는 주된 문제는 자원의 크기(공공 자전거의 경우 자전거와 거치대의 수)를 구하는 문제(fleet-sizing problem)와 자원을 재분배(공공 자전거의 경우 자전거 재배치)하는 문제(empty vehicle allocation problem)이다[9].

자전거 재배치는 한정된 자원을 이용하여 서비스를 극대화하기 위한 방안으로 두 가지 전략을 포함하고 있다. 첫 번째는 수요가 적은 대여소에 있는 여유 자전거를 수요가 많은 대여소로 이동시켜 이용자의 요구를 충족시키는 것으로 일반적인 대여 시스템에서 활용되는 전략이다. 두 번째는 반납이 많이 발생하는 대여소의 빈 거치대 확보를 위해 반납이 적게 발생하는 대여소로 자전거를 이동시키는 것이다. 이러한 자전거 재배치는 동적 재배치와 정적 재배치로 나뉜다[2, 3, 10]. 동적 재배치는 시스템의 운영 시간 중 어느 때에도 재배치를 수행할 수 있는 경우에 해당하고, 정적 재배치는 자전거 이용이 정지되어 있는 정해진 시간에만 재배치를 수행토록 하는 것이다. 일반적으로 하루 중 사용자가 없는 야간 시간대에 차량을 이용하여 다음 날의 서비스 향상을 위한 정적 재배치를 수행한다.

최근에 이루어진 정적 재배치에 대한 대부분의 연구는 두 가지 주제로 수행되어 왔다. 첫 번째는 재배치 결과 달성되어야 하는 각 대여소에서의 목표 재고 수준을 결정하는 것이고, 두 번째는 목표 재고 수준과 현 재고 수준을 입력하여 차량의 라우팅과 각 대여소에서의 자전거 상하차 수량을 결정하는 것이다. Sayarshad et al.[13]은 목표 재고 수준을 결정하기 위해 자전거 대여 수입과 비용(운영비용, 수송비용, 구매비용, 재고비용, 패널티 비용)을 고려하여 주어진 기간 동안의 총 수익을 최대화하는 MILP(Mixed Integer and Linear Program) 문제를 생성하였다. 그들의 연구에서는 시간에 따른 자전거 이용 수와 반납 수의 패턴 및 불확실성을 고려하지 않고, 정적인 특성을 갖는 주어진 기간 동안의 총 자전거 흐름량을 입력하여 하였다. 반면, Raviv and Kolka[11]는 각 대여소에서의 이용자 수와 이용 후 반납 수가 시간에 따라 서로 다른 패턴을 보이는 NHP(non-homogeneous Poisson) 과정에 따른다는 가정하에서 고객 불만족도를 최소화하는 목표 재고 수준을 구하는 방법을 제안하였다.

목표 재고 수준과 현 재고 수준을 입력하여 정적 재배치를 위한 두 가지 의사결정인 차량 라우팅과 대여소에서의 상하차 수량을 결정하기 위한 대부분의 연구들은 설정된 가정하에서 수학적 모델을 만들고, 이를 해결하는 알고리즘의 성능을 분석하는 절차로 이루어진다. Chelma et al.[3]은 목표 재고 수준을 만족하는 라우팅 중 가장 적은 운행시간을 갖는 라우팅을 구하기 위한 MILP(Mixed Integer and Linear Problem) 문제를 만들었다. 목표 수준을 맞추기 위해 각 대여소는 한 번 이상의 방문이 허용되도록 하였고, 차량은 한 대 만이 있는 경우를 대상으로 하여 Tabu search를 이용해 문제의 해를 구하였다. 목표 재고 수준을 꼭 맞추기 위한 제한 조건은 매우 엄격한 조건에 속하여 이를 만족하는 가능 해(feasible solution)를 구하기가 쉽지 않다. 이보다는 목표 재고 수준과 재배치 후 달성되는 재고 수준의 차이를 허용하여 이를 최소화하는 문제가 더 현실적일 수 있고, 가능 해를 보다 쉽게 구할 수 있는 장점을 갖는다. Raviv et al.[12]은 목표 재고 수준 미달성도와 차량의 운행시간을 최소화하는 MILP 문제를 만들었다. 다수의 차

량을 고려하였고, 차량은 각 대여소를 최대 한 번만 방문하도록 하였지만 여러 차량이 동일한 대여소를 방문할 수 있도록 하였다. 다수의 차량 라우팅과 상하차 수량을 구하여야 하는 문제의 복잡성으로 인하여 해를 효율적으로 구하기 위한 방법으로 휴리스틱과 완화(relaxation) 기법을 이용하였다. Gaspero et al.[5]은 Raviv et al.[12]의 연구에서와 같이 목표 재고 수준 미달성도와 차량의 운행시간을 최소화하는 N대의 차량 라우팅을 결정하는 문제를 만들었다. 기본적으로 각 대여소는 한 번만 방문토록 하였고, Large Neighborhood Search 방법으로 문제의 해를 구하였다. Rainer-Harbach et al.[10]은 목표 재고 수준 미달성도, 이동 자전거 수, 그리고, 차량 운행시간의 최소화를 위한 문제를 대상으로 GRASP(Greedy Randomized Adaptive Search Procedure), VNS (Variable Neighborhood Search) 등의 휴리스틱 방법으로 차량 라우팅을 구하고, 이에 부가하여 GH (Greedy Heuristic), MF-MC(Maximum Flow Approach for the Monotonic Case), LP(Linear Programming) 등의 방법으로 각 대여소에서의 상하차 수량을 구하였다. 이와 같이 목표 재고 수준과 현 재고 수준을 입력으로 하는 정적 재배치 문제들은 VRPPD(Vehicle Routing Problem with Pickup and Delivery)[7]와의 유사한 특성으로 인해 NP-hard 문제에 속하여 최적해 보다는 휴리스틱 방법[1, 2]에 의한 근사 최적해를 구한다.

본 연구에서는 정적 재배치에 대한 두 가지 주제인 목표 재고 수준과 차량의 라우팅 및 상하차 수량을 결정하는 문제 해결을 위해 실제 자료를 이용하여 실험한 결과를 제시한다. 목표 재고 수준은 시스템에서 보유하고 있는 총 자전거의 수를 고려하여 결정되어야 한다. 이를 위해 Raviv and Kolka[11]의 방법에 기반한 절차를 제시한다. 또한, 차량의 라우팅 및 상하차 수량을 결정하기 위해 기존 연구에서 개발된 복잡한 MILP 문제 보다는 함축적으로 표현된 문제를 대상으로 VNS를 적용한 결과를 소개한다. 각 대여소에서의 상하차 수량을 결정변수로 문제에 포함시키기 보다는 차량 라우팅에 함축적으로 포함되도록 하였다. 이는 문제의 복잡성을 보다 단순화하여 우수한 해를 효율적으로 구하기 위한 목적을 갖는다. VNS에서 필요로 하는 이웃해 구조는 VRP(Vehicle Routing Problem)에서 사용되는 구조와 동일하다. 따라서, VRP에 적용된 다양한 부분 탐색 방법[4]을 정적 재배치 문제에 그대로 사용할 수 있다. 이 중 단순성과 효율성의 장점을 가진 재위치(relocate) 기반의 연산을 사용하여 현 재해에서 부분해로의 랜덤 이동 방법인 Shaking 과 VND(Variable Neighborhood Decent)에 대한 세부적인 분석을 수행하였다. 특히, 대전광역시에서 운영하는 공공 자전거 대여시스템인 타슈를 대상으로 실제 이용 현황 자료를 분석하여 이용자의 수요 패턴을 모델링하고, 이들 자료를 바탕으로 한 실험을 통해 재위치 연산에 기반한 VNS의 성능을 분석하였다.

2. 문제 정의

본 연구에서 다루는 공용자전거 정적 재배치 차량 운행계획 문제는 다음과 같다. 네트워크 $G(V, A)$에서 $V=\{0, 1, 2, \cdots, n\}$는 노드 집합이고, $A=\{(i, j) : i \neq j\}$는 아크 집합이다. 노드 0은 관리 센터, 나머지 노드들은 대여소를 나타낸다. 관리센터를 제외한 각 노드는 거치대 수인 C_i와 초기 재고 수준 I_i가 설정되어 있고, 아크 (i, j)에는 재배치 차량의 운행시간 t_{ij}가 설정되어 있다. 현재 동일한 용량 Q를 갖는 m개의 빈 차량이 관리센터에 위치하고 있다. 이러한 네트워크에서 고객 불만족도와 차량 운행시간을 최소화하는 각 차량의 라우트, 그리고 각 대여소에서의 재배치 수량을 결정하는 것이 문제의 목적이다.

위 문제의 해 중 차량 라우트 $R=\{r, r_2, \cdots, r_m\}$은 m개의 라우트로 구성되고, 각 라우트는 차량에 의한 노드 방문 순서로 $r_k=(v_{k,1}\ v_{k,2}, \cdots, v_{k,n_k})$, $v_{k,j} \in V$이다. 각 라우트는 관리센터에서 시작하여 관리센터에서 종료된다.

문제에서 최소화하여야 하는 첫 번째 목적인 고객 불만족도는 어느 기간 동안 자전거가 부족하여 이용하지 못하는 고객 수 N_1과 거치대 부족으로 반납을 하지 못하는 고객 수 N_2의 함수로 표현될 수 있다.

자전거 부족으로 인해 한 고객이 경험하는 불만족 정도와 거치대 부족으로 인해 한 고객이 경험하는 불만족 정도를 각각 p, q라고 한다면 주어진 기간 동안의 총 불만족도는 다음과 같이 표현될 수 있다[7].

$$f = p \cdot N_1 + q \cdot N_2$$

f를 최소화하는 직접적인 결정변수는 각 대여소에서의 목표 재고 수준이다. 이는 차량에 의한 재배치 결과 달성되어야 하는 자전거 재고 수량으로 간주될 수 있다. 따라서, 고객 불만족도를 최소화하는 목표 재고 수준을 결정하였다면 차량 운행계획에서의 고객 불만족 최소화는 목표 재고 수준인 G_i의 달성도로 대체될 수 있다.

차량 라우트를 나타내는 해 $R=\{r_1, r_2, \cdots, r_m\}$는 각 대여소에서의 재배치 수량에 대한 의사 결정을 포함하고 있는 것으로 간주한다. 즉, 라우트 $R=\{r_1, r_2, \cdots, r_m\}$로 인해 재배치 후 얻어지는 대여소 i에서의 재고 수준 $Y_i(R)$은 다음과 같은 차량과 대여소 간의 자전거 상하차 규칙에 의해 결정된다고 가정한다. 라우트 R에 의해 대여소 i에 도착한 차량 k에 싣고 있는 자전거 수를 $I_{k,i}$라고 하고, 목표 재고 수준과 초기 재고의 차이를 $d_i = G_i - I_i$라고 하자. 하차, 상차되는 자전거의 수는 다음 두 규칙에 의한다.

1) 만약 $d_i \geq 0$이면 $\min\{d_i, I_{k,i}\}$개를 하차 한다.
2) 만약 $d_i < 0$이면 $\min\{-d_i, Q - I_{k,i}\}$개를 상차 한다.

이와 같이 결정된 재배치 후 재고 수준 $Y_i(R)$과 목표 재고 수준의 차이에 대한 절대값의 합은 목표 재고 수준의 미달성도로 이를 고객 불만족도의 척도로 이용한다.

$$Z_1(R) = \sum_{i=1}^{n} |Y_i(R) - G_i|$$

두 번째 목적인 차량 운행시간을 최소화하기 위한 목적함수는 단순히 모든 차량의 총 운행시간을 최소화하기 보다는 각 차량의 운행시간에 대한 최대값을 최소화하도록 한다. 이는 모든 차량이 운행을 종료하는 시각을 최소화하는 것으로 차량들의 운행시간 간 불균형을 최소화하는 목적을 갖는다.

$$Z_2(R) = \max_k \left\{ t(0, v_{k,1}) + \sum_{j=1}^{n_k - 1} t(v_{k,j}, v_{k,j+1}) + t(v_{k,n_k}, 0) \right\}$$

위의 두 목적함수에 가중치 w_1, w_2를 곱하여 최종적인 목적함수를 다음과 같이 정의한다.

$$Min\, Z = w_1 \cdot Z_1(R) + w_2 \cdot Z_2(R)$$

문제의 단순성을 위해 각 차량의 라우트는 일반적인 TSP(Traveling Salesman Problem), VRP(Vehicle Routing Problem)문제에서와 같이 각 대여소 노드 j가 한 번 만 방문된다는 제한 조건을 만족하도록 한다.

3. 목표 재고 수준 결정

3.1 마코브 체인에 기반한 기대 불만족 고객수 추정

이전에 정의된 정적 재배치 문제에서는 각 대여소에서의 목표 재고 수준 G_i에 대한 결정을 필요로 한다. 목표 재고 수준은 각 대여소에서의 불만족도를 최소화하는 목적에서의 재고수량이다. 어느 대여소에서 초기 재고가 I_0일 때 주어진 기간 동안 자전거가 부족하여 이용하지 못하는 고객 수 N_1과 거치대 부족으로 반납을 하지 못하는 고객 수 N_2는 I_0 함수로 표현될 수 있다. 불만족도 f를 최소화하는 최적해 I_0^*가 대여소에서의 목표 재고 수준으로 이를 구하기 위해서는 우선 $N_1(I_0)$와 $N_2(I_0)$의 값이 필요하다. 본 연구에서는 대여소에 이용자가 도착하는 수와 자전거가 반납되는 수가 NHP(non-homogeneous Poisson) 프로세스에 따르는 경우를 대상으로 하여 $N_1(I_0)$와 $N_2(I_0)$의 기대치를 구하기 위해 Raviv and Kolka[11]의 마코브 체인에 기반한 방법을 이용한다.

3.2 목표 재고 수준 결정 방법

주어진 기간 동안의 총 불만족도 F는 모든 대여소에서의 불만족도에 대한 합이다. F를 최소화하는 각 대여소에서의 목적 재고 수준($G_i,\ i=1, 2, \cdots, n$)을 구하는 문제를 다음과 같이 정의하였다.

$$Min \quad F=\sum_{i=1}^{n} f_i(G_i)$$

$$s.t) \ \sum_{i=1}^{n} G_i = S$$

$$G_i = 0, 1, 2, \cdots, C_i, \text{ for every } i$$

S는 재배치 대상이 되는 자전거의 총 수량이다. 첫 번째 제한 조건은 모든 대여소에서의 목표 재고 수준 합이 S가 되어야 한다는 것으로 S개의 자전거를 재배치하여 시스템 내에 위치하도록 하는 것이다. 두 번째 제한 조건은 각 대여소에서의 목표 재고 수준은 양의 정수로 거치대 수를 넘지 말아야 한다는 것을 의미한다.

만약 $f_i(G_i)$가 볼록함수라면 F를 최소화하는 각 대여소에서의 목표 재고 수준을 구하는 방법은 단순하다. 본 연구에서 개발된 방법은 다음과 같다.

1) 각 대여소에서 최소의 $f_i(G_i)$를 갖는 G_i를 초기해로 한다.

2) $\sum_{i=1}^{n} G_i = S$가 될 때까지 다음 절차를 반복한다.

   A. $\sum_{i=1}^{n} G_i < S$이면 $\Delta_k = \min_i \{\Delta_i = f_i(G_i+1) - f_i(G_i)\}$, where $G_i < C_i$인 대여소 k에 대해 $G_k = G_k + 1$

   B. $\sum_{i=1}^{n} G_i > S$이면 $\Delta_k = \min_i \{\Delta_i = f_i(G_i-1) - f_i(G_i)\}$, where $G_i > 0$인 대여소 k에 대해 $G_k = G_k - 1$

위 절차는 f_i가 볼록함수일 때 $\Delta_k \geq 0$이 되어 항상 최적해를 생성한다.

4. VNS 알고리즘

본 연구에서는 다양한 VNS 방법 중 예비 실험을 통해 정적 재배치 문제에 효과적이라고 판단되는 일반적인 VNS(general VNS)[4]라 일컫는 알고리즘을 적용하였다.

Algorithm: General VNS

0 Select the set of neighborhood structures N_k, for $k=1, \cdots, k_{max}$
Select the set of neighborhood structures N_l, for $l=1, \cdots, l_{max}$
find an initial solution R
choose a stopping condition

1 **Repeat** the following sequence until the stopping condition is met

1.1 $k=1$

1.2 **Repeat** the following steps until $k=k_{max}$

1.2.1 Shaking: Generate a point R' at random from the k th neighborhood $N_k(R)$ of R

1.2.2 Local search by VND

1.2.2.1 $l=1$

1.2.2.2 Repeat the following steps until $l=l_{max}$

1.2.2.2.1 Exploration of neighborhood: Find the best neighbor R'' of R' in $N_l(R')$

1.2.2.2.2 Move or not: If $f(R'')<f(R')$ set $R'=R''$ and $l=1$ otherwise $l=l+1$;

1.2.3 Move or not: If this local optimum is better than the incumbent, move there ($R=R''$), and continue the search with $N_l(k=1)$ otherwise, set $k=k+1$

정의된 정적 재배치 문제에서 해를 나타내는 m개의 차량 라우트는 일반적인 VRP에서 표현되는 해의 형태와 동일하므로 VRP에서 사용되는 부분 탐색 방법을 정적 재배치 문제에도 동일하게 적용할 수 있다. VRP 해결을 위해 다양한 부분 탐색 방법이 개발되었다[3]. 이 중 relocate에 기초한 부분 탐색 방법들을 수정하여 정적 재배치 문제에 적용하였다.

단계 0에서 생성된 초기해 $R=\{r_1, r_2, \cdots, r_m\}$는 각 대여소를 임의의 차량에 할당한 결과이다. 이 때 모든 차량에 가급적 동일한 수의 대여소가 할당되도록 함으로써 작업의 부하를 균등하게 하였다. 단계 1.2.1에서 Shaking을 통해 R의 이웃해인 R'을 생성하고 이를 입력으로 하여 단계 1.2.2에서 부분 탐색 방법에 기초한 VND를 적용한다.

부분 탐색을 위해 다음과 같은 두 가지 연산자를 정의한다. 해 R에서 대여소 v를 제거하는 연산을 $R-v$로 정의한다.

$$R-v=\{r_1, \cdots, r_{k-1}, r_k', r_{k+1}, \cdots, r_m\},$$

where $v_{k,p}=v$ and $r_k'=(v_{k,1}, \cdots, v_{k,p-1}, v_{k.p+1}, \cdots, v_{k,n_k})$

해 R에서 대여소 v를 k번째 라우트의 p번째 순서로 삽입하는 연산은 $[R+v]_{k,p}$로 정의한다.

$$[R+v]_{k,p}=\{r_1, \cdots, r_{k-1}, r_k', r_{k+1}, \cdots, r_m\},$$

where $r_k'(v_{k,1}, \cdots v_{k,p-1}, v, v_{k,p}, \cdots, v_{k,n_k})$.

다음에 설명될 두 가지 부분 탐색 방법인 1-relocate와 2-relocate는 현재 라우트에 위치한 대여소들을 다른 라우트의 다른 순서들로 이동시킨다.

4.1 1-relocate 부분 탐색에 기반한 VND

Algorithm: *1-Relocate*(R)

0 flag_1R = true task_1R = false
1 **while**(flag_1R = true) do
1.1 flag_1R = false
1.2 **for** each station i, $i=1, 2, \cdots, n$
1.2.1 $R^{(1)}=R-i$
1.2.2 find k^* and p^* such that
$z([R^{(1)}+i]_{k^*,p^*})=\min\{z([R^{(1)}+i]_{k,p}),$
for all k and $p\}$
1.2.3 **if** $z([R^{(1)}+i]_{k^*,p^*})<z(R)$,
then $R=[R^{(1)}+i]_{k^*,p^*}$ flag_1R = true
task_1R = true
end if
end for
end while
2 **return** task_1R

1-Relocate는 각 대여소에 대해 단계 1.2.1에서 현재의 해에서 삭제한 후 단계 1.2.2와 1.2.3에서 best-accept[1]에 기초하여 가장 큰 향상을 가져오는 k^*번째 차량의 라우트에서 p^*번째 방문순서로 이동을 고려한다. 이전의 해와 비교하여 목적 함수의 향상이 있을 경우에만 이동을 수행한다. 이러한 절차는 더 이상의 향상이 없을 때까지 반복된다.

4.2 2-relocate 부분 탐색에 기반한 VND

Algorithm: *2-Relocate*(R)

0 flag_2R = true; task_2R = false
1 **while**(flag_2R = true) do
1.1 flag_2R = false
1.2 **for** each station i, $i=1, 2, \cdots, n-1$
1.2.1 **for** each station j, $j=i+1, i+2, \cdots, n$
1.2.1.1 k1 is index of route in which station i is included
1.2.1.2 k2 is index of route in which station j is included
1.2.1.3 **if** k1 = k2, continue
end if

1.2.1.4 $R^{(1)} = R - i$

1.2.1.5 $R^{(2)} = R^{(1)} - j$

1.2.1.6 $R^{(3)} = [R^{(2)} + i]_{k2,\,p^*}$

where $z([R^{(2)} + v]_{k2,p^*})$

$= \min\{z([R^{(2)} + v]_{k2,p}),\ \text{for all}\ p\}$

1.2.1.7 $R^{(4)} = [R^{(3)} + j]_{k1,p^*}$

where $z([R^{(3)} + v]_{k1,p^*})$

$= \min\{z([R^{(3)} + v]_{k1,p}),\ \text{for all}\ p\}$

1.2.1.8 **if** $z(R^{(4)}) < z(R)$, then $R = R^{(4)}$

flag_2R = true task_2R = true

end if

end for

end for

end while

2 **return** task_2R

2-Relocate는 모든 두 대여소에 대해 현재의 라우트에서 각각 상대 대여소가 속한 라우트로의 이동을 수행한다. 이 때 가장 큰 향상을 가져오는 방문 순서로의 이동을 고려하고, 이전 해와 비교하여 목적함수의 향상이 있는 경우에 이동을 수행한다. 1-Relocate와 마찬가지로 이러한 절차는 더 이상의 향상이 없을 때까지 반복된다. 1-relocate에서는 각 라우트에 포함된 대여소 수가 변경되는 반면, 2-relocate에서는 각 라우트에 포함된 대여소 수가 변하지 않는다.

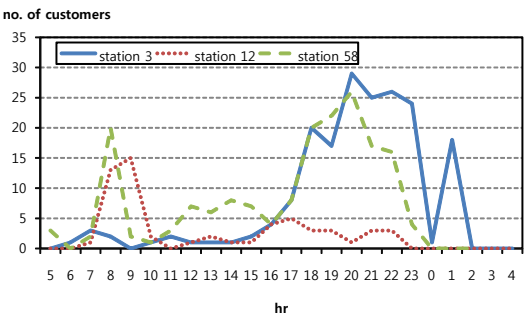
현재해 R에 대한 shaking으로 다양한 연산을 적용할 수 있다. Rainer-Harbach[9] 등은 라우트의 일부 순서를 이동, 교체하거나 한 대여소를 라우트에서 제거하고 다른 라우트에 삽입하는 연산 등을 적용하였다. 본 연구에서는 shaking의 강도에 대한 효과를 분석하려는 목적에서 임의의 두 대여소를 선택하여 라우트와 방문순서를 서로 교환하는 단위 연산을 반복 수행토록 하였다. 이러한 교환연산의 횟수의 크기에 따라 Shaking의 강도가 다르게 된다. 교환 횟수가 클수록 현재해 R과 VND에 적용되는 초기해 R'의 차이가 크게 되어 shaking의 강도가 큰 효과를 가져온다.

5. 실험 및 분석

대전시에서 운영하는 타슈 공공 자전거 시스템에는 2013년 6월 3일 현재 114개의 대여소가 있고, 각 대여소는 9개부터 28개 사이의 거치대를 가지고 있다. 재배치를 위한 차량은 5대가 있으며 용량은 20이다. 일일 실적자료를 기초로 정적 재배치 문제를 생성하여 VNS 알고리즘의 성능 분석을 위한 실험을 수행하였다. 알고리즘은 자바 프로그래밍 언어로 코딩되어 2.5 GHz의 PC에서 실행되었다.

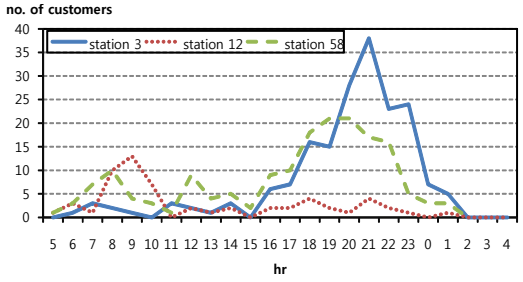
5.1 목표 재고 수준 결정

각 대여소에서의 목표 재고 수준은 익일의 시간별 자전거 이용 고객 수와 반납 고객 수의 예측에 따라 결정되어야 한다. 부분 탐색 알고리즘의 성능을 분석하는 실험에서는 타슈의 일일 실적자료로 예측자료를 대신하도록 한다. 2013년 6월 3일 실적 자료를 분석한 결과 고객 도착율과 반납율은 대여소 마다 서로 다른 패턴을 보이고 있다. [그림 1]은 6월 3일 오전 5시에서 익일 5시까지의 3개 대여소에 대한 시간 당 고객 도착수를 나타낸다. 12번 대여소는 오전 7시부터 증가하기 시작하여 9시 피크를 이루어 출근 시간대 이용자가 많은 반면 3번 대여소는 오전에는 사용자가 거의 없고 오후 8시 대에 피크를 이루어 퇴근 이후에 이용자가 많다. 58번 대여소는 오전 8시와 오후 8시에 피크를 이루고 오후 이용자수가 오전 보다 다소 많은 패턴을 보인다.



[그림 1] 3개 대여소에서의 시간 당 고객 도착율

반납율 또한 대여소와 시간대에 따라 상이한 패턴을 보인다. 3개 대여소에서의 시간별 반납 수를 나타내는 [그림 2]에서 3번 대여소는 오후 반납자가 많고, 12번 대여소는 오전 반납자가 많다. 58번 대여소는 오전 8시와 12시, 그리고 오후 8시에 피크를 이룬다.

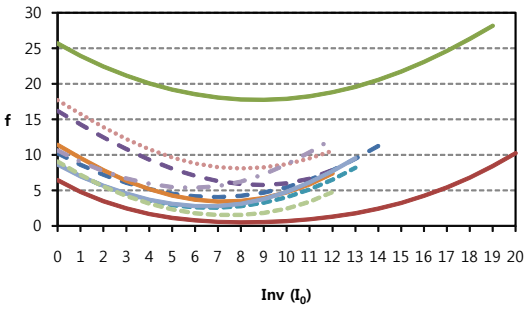


[그림 2] 3개 대여소에서의 시간 당 자전거 반납율

각 대여소에서의 시간별 도착율과 반납율은 NHP 분포에 따르는 것으로 가정하였다. 114개 대여소에서의 시간대별 도착율과 반납율을 이용하여 제3.1절에 설명된 초기 재고에 따른 기대 불만족 수를 구하기 위해 다음과 같이 설정하였다.

- 오전 5시부터 익일 2시까지 21시간 동안을 대상으로 하였다.
- 증가시각을 1분으로 하여 전이확률과 상태확률을 구하였다.
- 상태확률을 바탕으로 21시간 동안의 $E[N_1(I_0)]$과 $E[N_2(I_0)]$를 구하였다.

마코브 체인에 기반한 방법에 의해 구한 각 대여소에서의 불만족도 $f = p \cdot E[N_1(I_0)] + q \cdot E[N_2(I_0)]$는 Raviv and Kolka[10]의 연구에서 보인 것과 동일한 결과로 볼록함수에 따르는 것으로 나타났다. [그림 3]은 p와 q의 값을 1로 하였을 때 임의의 10개 대여소에서 초기 재고량 I_0에 따른 f를 나타낸다. 어느 대여소에서도 f가 볼록함수의 성질을 가지는 것을 알 수 있다. 이 같은 특성은 제3.2절에서 설명된 목표 재고 수준을 결정하는 방법의 적용이 유효함을 나타낸다. 이에 따라 총 재고 수량 S를 900으로 하여 각 대여소에서의 목표 재고 수량을 결정하였다.



[그림 3] 10개 대여소에서의 초기 재고에 따른 고객 불만족도

5.2 초기 재고 수준 생성

각 대여소에서 목표 재고 수준과 초기 재고 수준이 동일하다면 더 이상의 재배치 작업은 필요치 않다. 반면 두 수량의 차이가 클수록 더 많은 재배치 작업이 필요하게 된다. 재배치 작업량에 따른 부분 최적화 방법의 성능을 분석하기 위해 각 대여소에서의 초기 재고는 다음과 같이 설정하였다. 우선 각 대여소에 대해 목표 재고 수준과 초기 재고의 차이인 $d_i = G_i - I_i$가 균등분포에 따르도록 초기 재고를 설정한다. 이 때 초기 재고의 합과 목표 재고의 합은 시스템에 있는 총 자전거수와 일치하도록 한다. 이렇게 구한 초기 재고를 입력으로 하여 다음 절차를 l번 반복한다.

1. 임의의 서로 다른 두 대여소 i, j를 선택한다. 이 때 대여소 i는 초기 재고 수준이 목표 재고보다 작거나 같고, 대여소 j는 반대로 초기 재고 수준이 목표 재고보다 크거나 같도록 선택한다.
2. 선택된 두 대여소에 대해 대여소 i는 초기 재고 수준을 하나 줄이고 대여소 j는 초기 재고 수준을 하나 증가시킨다. 이에 따라 두 대여소에서 필요한 재배치 작업량이 증가하게 된다.

반복 횟수 l이 증가함에 따라 요구되는 재배치 작업은 더욱 많아질 것이다. 전체 114개 타슈 대여소를 대상으로 한 재배치 문제에서는 l의 값을 0(LOW), 50(MED), 100(HIGH)으로 하여 실험을 위한 문제에서의 초기 재고를 결정하였다.

5.3 VND에서의 부분 탐색 알고리즘 성능비교

각 대여소에서 결정된 목표 재고 수준과 초기 재고 수준을 입력으로 정적 재배치 문제의 해를 구하는 VND 알고리즘의 성능을 분석하기 위한 실험을 수행하였다. 입력 데이터 중 대여소 간 차량 운행시간은 각 대여소의 좌표(경도, 위도)를 바탕으로 한 유클리디안 거리로 대체하였고, 용량 20의 5대 재배치 차량을 대상으로 하였다. 정의된 정적 재배치 문제의 목적함수 Z에 포함된 w_1과 w_2는 각각 목표 재고 수준의 미달성도 Z_1과 최대 차량 운행시간 Z_2에 대한 가중치이다. 제5.5절에서 수행된 바와 같은 가중치 결정을 위한 실험을 통해 0.6, 0.4를 각각 w_1과 w_2의 값으로 설정하였다.

<표 1>은 재배치 작업량 LOW, MED, HIGH의 문제를 대상으로 실험한 결과이다. 각 문제에 대해 50개의 초기해를 임의로 구하였고, 이들 각 초기해에 대해 1-relocate 만을 적용한 VND와 2-relocate 만을 적용한 VND의 두 가지 VND를 적용하였다. 표에서 각 문제에 대한 수치는 이러한 50번의 반복 실험을 통한 결과인 목적함수 값의 평균(ave)과 표준편차(std), 그리고 일회 실행시간(cpu)을 의미한다. <표 1>은 모든 문제에 대해 VND 알고리즘에 의한 해의 질이 매우 우수함을 나타낸다. LOW 문제에 대한 초기해 값의 평균이 5529.3인 것에 비해 1-relocate는 2790.1로 50%의 감소를 가져왔고, 2-relocate는 1229.9로 78%의 감소를 가져와 1-relocate에 비해 2-relocate가 더 우수한 해를 생성하였다. 각 문제에 대한 50개 해의 값에 대한 표준편차 역시 2-relocate가 136.3으로 1-relocate의 613.7보다 4.5배 정도 적어 보다 안정적으로 우수한 해를 생성한다. 반면 1-relocate 해들의 표준편차는 초기해 보다도 커 더욱 편차가 큰 해들을 생성하는 불안정적인 특성을 나타낸다.

LOW 문제뿐만 아니라 MED, HIGH의 문제에서도 이와 유사한 결과를 가져온다. 실행시간 측면에서는 1-relocate가 평균 0.1초인 것에 비해 2-relocate는 36.2초로 329배 정도 더 많은 실행시간이 걸렸다. 1-relocate는 2-relocate에 비해 탐색 공간이 적어 매우 빠르게 부분 최적해에 수렴한 것으로 분석된다. 2-relocate 방법은 1-relocate에 비해 안정적으로 매우 우수한 해를 생성한다.

5.4 VNS 알고리즘에서 Shaking 강도 분석

이전의 분석에서 부분 탐색 알고리즘의 성능이 매우 우수함을 보였다. 이에 따라 일반적 VNS에 포함된 VND에서 이 두 부분 탐색 알고리즘을 순차적으로 적용하였다. 2-relocate가 1-relocate에 비해 더 우수한 결과를 가져오므로 2-relocate를 먼저 수행하고, 그 후에 1-relocate를 수행하였다. 즉, General VNS 알고리즘에서 l_{max}를 2로 하여 $N_1(k)$는 2-relocate 알고리즘, $N_2(k)$는 1-relocate 알고리즘으로 하였다.

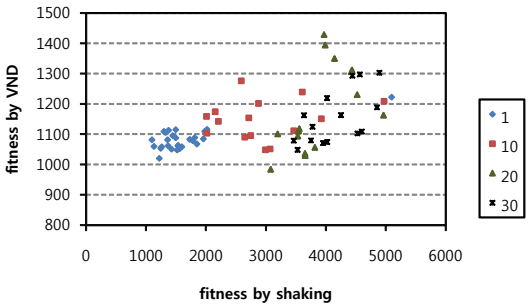
VNS 알고리즘에서 현재해에 shaking을 가한 해를 초기해로 하여 VND를 수행한다. Shaking 의 강도를 약하게 하면 현재해에서 가까운 이웃해 만을 생성하기 때문에 탐색 공간이 좁아지는 반면 보다 세밀한 탐색을 수행할 수 있다. 그러나, shaking의 강도를 세게 하면 현재해에서 멀리 떨어진 해를 초기해로 하여 탐색 공간이 넓어질 수 있으나 세밀한 탐색이 이루어 지지 않는 단점이 있다. 본 연구에서는 shaking에서의 이웃해 생성 방법을 두 대여소의 교환으

〈표 1〉 VND에서의 부분 최적화 방법 비교

Problem	Initial		1-relocate			2-relocate		
	ave	std	ave	std	CPU(sec)	ave	Std	CPU(sec)
LOW	5529.3	451.1	2790.1	613.7	0.1	1229.9	136.3	36.2
MED	5408.1	357.0	2766.3	564.5	0.1	1197.6	71.4	35.6
HIGH	5413.3	448.6	2966.9	703.0	0.1	1194.8	93.8	36.0
ave	5450.2	418.9	2841.1	627.1	0.1	1207.4	100.5	35.9

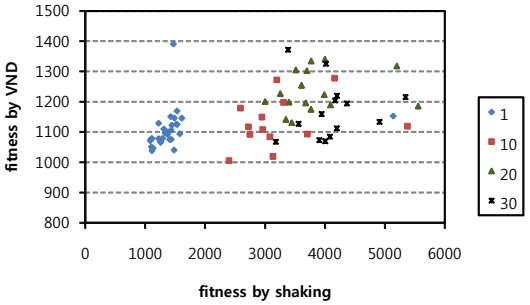
로 하였다. 즉, 현재해에서 임의의 두 대여소를 선택하여 라우트와 방문순서를 서로 교환하는 것이다. Shaking의 강도에 따른 VNS 알고리즘의 성능을 분석하기 위해 이러한 두 대여소의 교환 횟수를 1, 10, 20, 30의 4가지로 하여 각 교환 횟수에서 알고리즘을 실행하였다. 단, 알고리즘의 시작 단계에서 생성하는 최초의 현재해는 교환 횟수에 상관없이 모두 동일한 해로 하여 공정한 비교가 이루어지도록 하였다. VNS의 종료 조건인 알고리즘 실행시간은 2-relocate 부분 탐색 방법을 20회 정도 실행하는 시간인 10분으로 정하였다.

[그림 4]는 LOW 문제에 대해 4가지 shaking 강도에서 VNS를 실행한 결과를 나타낸다. VNS 알고리즘이 실행되는 동안 각 Shaking 강도에서 현재해로부터 생성되는 해와 그 해를 초기해로 하여 VND를 수행한 부분 최적해들의 목적함수 값들의 관계를 나타낸다.

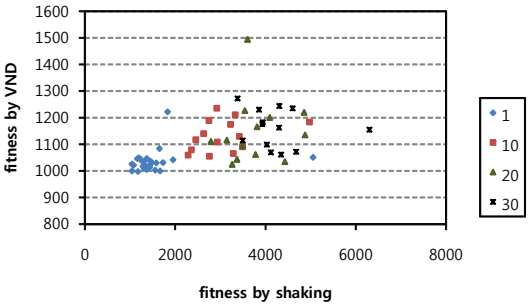


[그림 4] LOW 문제에서 Shaking 강도의 효과

Shaking의 강도가 커질수록 VND의 초기해들은 큰 편차를 보이고 있어 현재해에서 멀리 떨어진 해를 생성하고, 부분 최적해 역시 편차가 큰 결과를 나타낸다. 대여소 교환을 1회 만 한 경우에는 현재해에서 아주 가까운 이웃해를 생성하고, 이렇게 가까운 이웃해들을 초기해로 한 부분 최적해 역시 이전의 부분 최적해와 크게 다르지 않는 특성을 보인다. Shaking 강도가 큰 경우 VND를 통한 부분 최적해들이 Shaking의 강도가 적은 경우에 비해 목적함수 값이 감소되었다고 있다고 볼 수 없다. MED, HIGH 문제에 대한 결과인 [그림 5]와 [그림 6]에서도 동일한 결과가 도출된다. Shaking의 강도를 적게 하여 좁은 지역을 세밀하게 탐색하는 것이 보수적인 방법이긴 하지만 우수한 해를 생성할 수 있는 가능성이 높음을 의미한다. 반면 Shaking의 강도를 크게 하면 넓은 탐색 공간으로 하여 전역 최적해를 구할 수 있는 가능성이 있을지 모르지만 탐색 공간의 크기와 한정된 실행시간을 감안하면 위험이 큰 방법이라고 할 수 있다.



[그림 5] MED 문제에서의 Shaking 강도 효과



[그림 6] HIGH 문제에서 Shaking 강도 효과

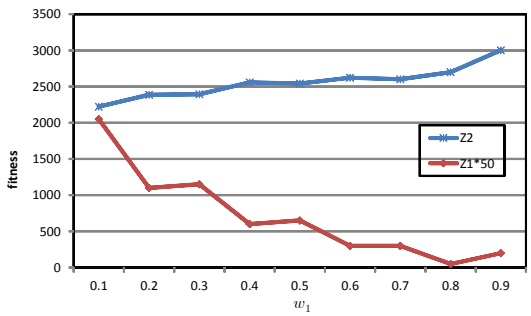
5.5 가중치 결정

정의된 정적 재배치 문제는 목적함수 Z에 포함된 목표 재고 수준의 미달성도(Z_1)에 대한 가중치와 최대 차량 운행시간(Z_2)에 대한 가중치의 결정을 필요로 한다. [그림 7]은 MED 문제에 대해 목표 재고 수준 미달성도에 대한 가중치 w_1을 0.1부터 0.9까지 0.1 단위로 증가시켜 VNS에 의해 생성된 해의 Z_1값을 50배한 값과 Z_2값을 나타낸다. 두 목적함수 값이 정규화되어 있지 않았지만 두 가중치의 합은 1이 되도록 하였다. VND에서는 이전의 실험에서와 같이 2-re-

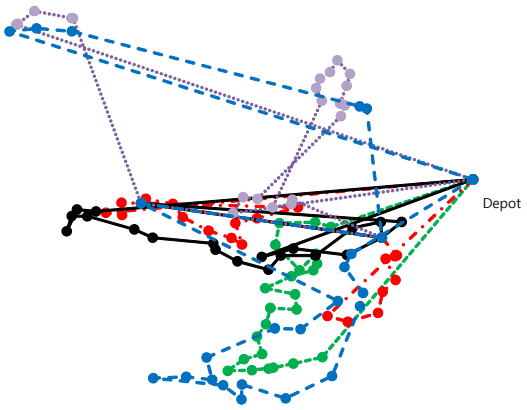
locate와 1-relocate를 순차적으로 실행하였고, shake는 1회의 대여소 교환을 수행하도록 하는 VNS 방법을 대상으로 하였다.

[그림 7]은 w_1이 클수록 목표 재고 수준의 미달성도를 나타내는 Z_1은 작아지고, 최대 차량운행시간 Z_2는 커져 두 양 사이에 상충효과가 존재하는 자연스러운 결과를 나타낸다. w_1이 0.1일 때 Z_1은 41로 재배치 후 목표 수량에 미달하거나 과다하게 되는 수량이 총 재고 수량 900의 4.6%에 해당한다. 그러나, 차량 라우트 들의 최대 운행시간은 2,221로 [그림 8]에서와 같이 각 트럭의 라우트는 비교적 잘 클러스터링된 대여소들을 포함하고 있다. 반면, w_1이 0.9일 때 Z_1은 4가 되어 모든 대여소에서의 재배치 후 재고 수량은 목표 재고수량과 거의 일치하나 [그림 9]에서와 같이 비교적 클러스터링 되지 않은 차량 라우트로 인해 최대 운행거리는 3,001이 되어 w_1이 0.1일 때의 2,221에 비해 35%의 최대 운행시간이 증가되었다.

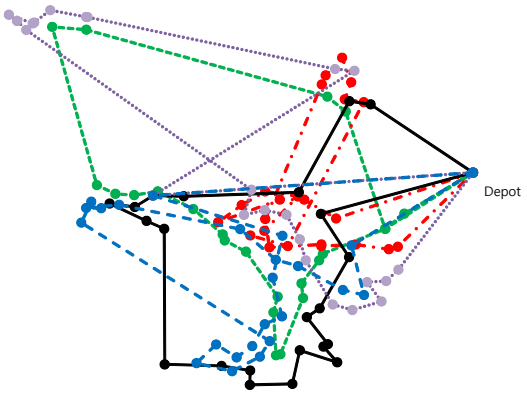
두 수준의 한계 설정은 가중치 결정에 큰 도움을 준다. [그림 7]에서 최대 운행시간의 한계를 2,600으로 설정한다면 w_1은 0.5 이하이어야 한다. 또한, 목표 재고 수준을 맞추지 못하는 수량을 최대 12로 제한한다면 w_1은 0.4 이상이어야 한다. 두 한계를 동시에 설정한다면 결국 w_1은 0.4에서 0.5 사이의 값이 된다. 차량의 최대 운행시간은 시스템의 운영 비용과 밀접한 관계가 있고, 목표 재고 수준의 미달성도는 서비스 비용과 관계가 있다. 이 두 비용을 고려한 두 수준의 한계를 설정하는 것이 필요하다.



[그림 7] 가중치에 따른 목적함수 값



[그림 8] MED 문제에서 w_1이 0.1일 때 VNS에 의한 차량 라우트



[그림 9] MED 문제에서 w_1이 0.9일 때 VNS에 의한 차량 라우트

6. 결 론

공공 자전거 시스템에서의 정적 재배치는 자전거 사용이 발생하지 않는 야간 시간대에 트럭을 이용하여 자전거를 재배치하는 것으로 자전거와 거치대가 부족하여 이용과 반납을 하지 못하는 사용자의 불만족을 최소화하는 것이 일차적인 목적이다. 본 연구에서는 정적 재배치를 위해 고객의 불만족을 최소화하는 목적에서 각 대여소에서의 목표 재고 수량을 결정하였고, 목표 재고 수준의 불만족 정도와 차량의 운행시간 최소화를 위한 각 차량 라우트와

각 대여소에서의 자전거 상하차 수량을 도출하기 위해 VNS 메타 휴리스틱 알고리즘을 적용하였다. 특히, 일반적 VNS에 포함되는 VND 알고리즘으로 relocate에 기반한 부분 탐색 알고리즘을 적용하였다. 대전 타슈 시스템의 실제 실적자료를 바탕으로 실험한 결과 1-relocate에 비해 2-relocate 부분 탐색 방법이 실행시간의 증가 부담이 있었지만 더 안정적이고 우수한 해를 생성하였다. 특히, 2-relocate와 1-relocate를 결합하여 순차적으로 적용한 VND 방법으로 해의 향상을 가져올 수 있었다. VND의 초기해를 생성하는 shaking 으로는 현재해에서 임의의 두 대여소를 교환하는 방법을 적용하였다. 교환 횟수에 따라 shaking의 강도가 달라지고, 이에 따라 현재해와 생성되는 이웃해의 거리가 달라진다. Shaking의 강도가 작은 경우에는 가까운 이웃해만을 생성하여 탐색 공간이 작아지는 반면 세밀한 탐색을 수행할 수 있고, Shaking의 강도가 큰 경우에는 탐색 공간이 커지는 반면 세밀한 탐색이 이루어지지 못한다. 대전 타슈의 경우에는 실행시간의 제약과 부분 탐색 방법의 우수성으로 인해 Shaking의 강도를 적게 하는 경우가 보다 안정적인 좋은 해를 생성할 수 있음을 보였다. 본 논문에서 제안하고 사용한 방법이 GA, PSO 등의 다른 메타 휴리스틱 또는 다른 알고리즘과 비교하여 어떤 성능 차이를 가지는지에 대한 후속 연구가 필요할 것으로 예상한다. 이는 타슈 데이터 외 다양한 시스템에서의 데이터를 입력으로 성능 분석이 이루어질 것으로 기대한다.

참 고 문 헌

[1] 윤태용, 이상헌, “주기적 다용량 차량경로 문제에 대한 발견적 해법”, 「한국경영과학회지」, 제36권, 제1호(2011), pp.27-38.

[2] 이상헌, 이승원, “시간제약이 있는 차량경로 문제에 대한 개미군집 시스템해법”, 「한국경영과학회지」, 제34권, 제1호(2009), pp.153-165.

[3] Chemla, D., F. Meunier, and R.W. Calvo, “Bike Sharing Systems : Solving the Static Rebalancing Problem,” *Discrete Optimization*, Vol.10 (2013), pp.120-146.

[4] Groer, C., B. Golden, and E. Wasil, “A Library of Local Search Heuristics for the Vehicle Routing Problem,” *Math. Prog. Comp.*, Vol. 2(2010), pp.79-101.

[5] Gaspero, L.D., A. Rendl, and T. Urli, “Constraint-Based Approaches for Balancing Bike Sharing Systems,” *LNCS*, Vol.8124(2013), pp.758-773.

[6] Hansen, P., N. Mladenovic, and J.A.M. Perez, “Variable Neighborhood Search : Methods and Applications,” *Ann Oper Res*, Vol.175 (2010), pp.367-407.

[7] Hernandez-Perez, H. and J. Solazar-Gonzalez, “A Branch-and-Cut Algorithm for a Traveling Salesman Problem with Pickup and Delivery,” *Discrete Applied Mathematics*, Vol.145(2004), pp.126-139.

[8] Kloimullner, C., P. Papazek, B. Hu, and G.R. Raidl, “Balancing Bicycle Sharing Systems : An Approach for the Dynamic Case,” *LNCS*, Vol.8600(2014), pp.73-84.

[9] Kochel, P., S. Kunze, and U. Nielander, “Optimal Control of a Distributed Service Systems with Moving Resources : Application to the Fleet Sizing and Allocation Problems,” *International Journal of Production Economics*, Vol. 81(2003), pp.443-459.

[10] Rainer-Harbach, M., P. Papazek, G.R. Raidl, B. Hu, and C. Kloimullner, “PILOT, GRASP, and VNS approaches for the static balancing of bike sharing systems,” *Journal of Global Optimization*, (2014), pp.1-33.

[11] Raviv, T. and O. Kolka, “Optimal Inventory Management of a Bike-Sharing Station,” *IIE Transactions*, Vol.45, No.10(2013), pp.1077-

1093.

[12] Raviv, T., M. Forma, and I.A. Tzur, "Static Repositioning in a Bike-Sharing System : Models and Solution Approaches," *EURO J. Trasnp Logist*, Vol.2, No.3(2013), pp.187-229.

[13] Sayarshad, H., S. Avassoli, and F. Zhao, "A Multi-periodic Optimization Formulation for Bike Planning and Bike Utilization," *Applied mathematical Modeling*, Vol.36(2012), pp.4944-4951.