유전자 알고리즘을 활용한 차량 선적 최적화 기법

차주형* · 우영운

동의대학교 응용소프트웨어공학과

A Method for Vehicle Loading Optimization Using Genetic Algorithm

Joo Hyoung Cha* · Young Woon Woo

Dept. of Applied Software Eng., Dong-eui University

E-mail : aoikazto@naver.com / ywwoo@deu.ac.kr

요 약

이 논문에서는 공간에 할당되는 데이터의 변환 및 시각화부터 유전자 알고리즘을 이용하여 차량 선적을 최적화하기 위한 기법을 제안하였다. 기존 방식의 유전자 알고리즘에서 교차, 변이, 우수한 유전자 생존, 대표 유전자 추출 과정들 중 교차 과정을 제외한 기법들을 활용하였다. 또한, 차량의 선적 공간을 병합, 분할하는 기법을 함께 활용하여 차량 선적 최적화 기법을 제안하였다. 실험 결과 기존의 유전자 알고리즘에서 사용되는 교차 기법으로 처리하기 힘든 부분에서 이 논문에서 제안한 병합, 분할 기법을 적용하는 것이 최적화 과정에 효과가 있음을 확인할 수 있었다.

ABSTRACT

In this paper, we proposed a technique for optimizing vehicle loading using genetic algorithms from the transformation and visualization of data allocated to space. In the conventional genetic algorithm, techniques for mutation, good gene survival, and representative gene extraction were used. In addition, the vehicle loading optimization technique is proposed by utilizing the merging and branching techniques of vehicle loading space. Experimental results show that the merging and dividing technique proposed in this paper is effective in the optimization process, which is difficult to handle with the crossover technique used in the existing genetic algorithm.

키워드

Genetic algorithm, Vehicle loading optimization, Merge and branch, Visualization

Ⅰ. 서 론

국제 무역이 활발해짐에 따라 하나의 차량 선적화물선이 여러 항구를 거쳐 일주하면서 대량의 차량을 선적하고 하적하는 업무가 빈번히 발행하고 있다. 하지만 선적하고자 하는 차량의 종류가 다양하고 선적, 하적이 항구마다 빈번히 일어남에 따라 선박의 어떤 공간에 어떤 목적지의 차량을 어떤 배치로 선적해야 하는 것이 복잡한 업무이다. 현재는 이 분야에 오래 근무한 전문가가 업무 경험을 통해 차량 선적, 하적 계획을 장시간에 걸쳐 수립하고 있다.

하지만 무역 구조가 복잡해짐에 따라 선박의 운항 도중에도 새로운 선적 요구와 하적 요구 사항이 발행하는 등 차량의 배치를 최적화하기 어려운 장애물이 많아 전문가라고 할지라도 차량 선적 계획을 체계적이고 효율적으로 하기가 어려운 현실이다[1].

이 논문에서는 이러한 차량 선적 계획을 인공지능 알고리즘을 통해 최적화하는 기법을 제안하고 구현하기 위한 아이디어를 제안한다.

* speaker

II. 데이터 분석

데이터의 차량을 선적을 하기 위한 공간의 크기, 접촉해서는 안 되는 부분의 영역 등에 대한 데이터가 xml 형식[2]으로 입력된다. 각 데이터들은 선박의 벽면 형태와 각 층으로 이동을 하는 경로와 같은 상세한 데이터가 폴리곤(polygon) 형식으로 데이터가 입력된다. 폴리곤으로 입력된 데이터를 x, y 좌표(미터 단위) 형식으로 입력이 되는데 이 좌표를 따라 시각화를 진행하면 선박 형태를 시각화할 수 있다. 샘플로 제공되는 데이터를 활용하여 차량을 선적할 수 있는 최대 공간을 탐색한다. 1AEU의 크기는 4.115m x 1.62m 크기의 공간을 의미한다. 선박에서는 1AEU 단위를 활용하여 필요 공간을 할당하고, 차량을 배치한다.

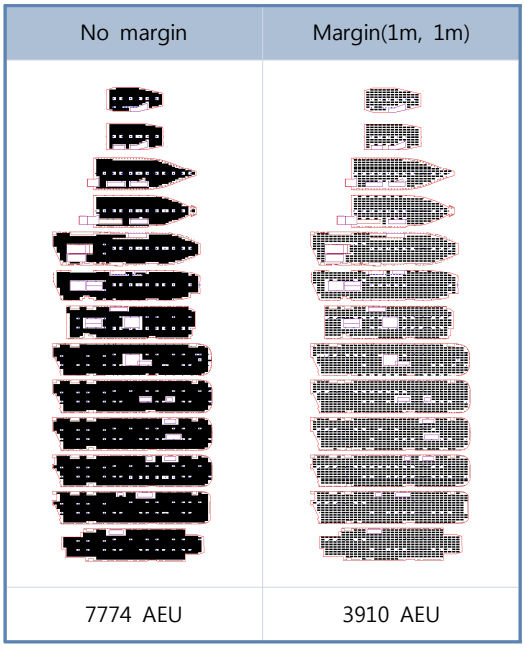

그림 1. Obstacle 영역을 제외한 공간에 대한 최대 AEU 계산

그림 1에서 알 수 있듯이 1AEU에 margin을 1m 단위로 계산하게 되면 빈 공간이 많아져서 선적 가능 영역이 절반 정도로 줄어드는 것을 알 수 있다.

다음으로 그림 2에 나타나 있는 것처럼 선박의 운항 경로 및 차량의 POL(Port Of Loading), POD(Port Of Discharge)[3] 데이터를 모은 뒤 정렬하여 차량이 몇 대인지가 아니라, 차량의 공간에 따른 필요 공간량을 계산하는 알고리즘을 구현하기 위해 필요한 데이터만 분석하기 위하여 Excel로 생성되어 있는 자료를 xml 파일로 변환하였다.

```
<Squence>
  <Port CountryCode="KR" POL="True" POD="False">KRUSN</Port>
  <Port CountryCode="CN" POL="True" POD="False">CNXGG</Port>
  <Port CountryCode="KR" POL="True" POD="False">KRPTK</Port>
  <Port CountryCode="KR" POL="True" POD="False">KRKUV</Port>
  <Port CountryCode="KR" POL="True" POD="False">KRMOK</Port>
  <Port CountryCode="KR" POL="True" POD="False">KRKAN</Port>
  <Port CountryCode="KR" POL="True" POD="False">KRMAS</Port>
  <Port CountryCode="KR" POL="True" POD="False">KRINC</Port>
  <Port CountryCode="PH" POL="False" POD="True">PHBTG</Port>
  <Port CountryCode="SG" POL="True" POD="True">SGSIN</Port>
  <Port CountryCode="ZA" POL="False" POD="True">ZADUR</Port>
  <Port CountryCode="ZA" POL="False" POD="True">ZAPLZ</Port>
  <Port CountryCode="PH" POL="False" POD="True">PHDAK</Port>
  <Port CountryCode="ZA" POL="False" POD="True">ZAVIG</Port>
  <Port CountryCode="ZA" POL="False" POD="True">ZABRS</Port>
  <Port CountryCode="FR" POL="False" POD="True">FRLEH</Port>
  <Port CountryCode="BE" POL="False" POD="True">BEANR</Port>
  <Port CountryCode="BE" POL="False" POD="True">BEZEE</Port>
  <Port CountryCode="DE" POL="False" POD="True">DEBRV</Port>
</Squence>
```

그림 2. 선박 운항 경로

```
<Port-Info>
  <POL>KRPTK</POL>
  <Arrives>
    <POD TotalAeu="195">PHBTG</POD>
    <POD TotalAeu="116">SGSIN</POD>
    <POD TotalAeu="344">PHDAK</POD>
    <POD TotalAeu="358">FRLEH</POD>
  </Arrives>
</Port-Info>
<Port-Info>
  <POL>KRKUV</POL>
  <Arrives>
    <POD TotalAeu="292">SGSIN</POD>
    <POD TotalAeu="537">ZADUR</POD>
    <POD TotalAeu="14">PHDAK</POD>
  </Arrives>
</Port-Info>
<Port-Info>
  <POL>KRMOK</POL>
  <Arrives>
    <POD TotalAeu="44">PHBTG</POD>
    <POD TotalAeu="33">SGSIN</POD>
    <POD TotalAeu="37">FRLEH</POD>
  </Arrives>
</Port-Info>
```

그림 3. 선박의 POL, POD 데이터

이동 항해 경로에 따른 필요 AEU와 POL 기준으로 먼저 항구에 가는 곳 기준으로 정렬하였으며 POL 내에서도 POD로 가장 근접한 곳에 하역하는 곳 기준으로 정렬하였다. 그림 3과 같이 정렬한 후 사람이 알아보기 쉽게 xml 데이터로 표현하고 관련 프로토콜을 정의하였다.

III. 분석 및 알고리즘 구현

공간에 대한 차량 선적을 최적화하기 위하여 딥러닝을 사용을 시도하였으나 샘플 데이터의 부족으로 인하여 적합도(fitness) 함수를 작성하여 최적의 모습을 찾아가는 알고리즘인 유전자 알고리즘을 선택하게 되었다[4]. 차량 선적을 최적화를 1AEU 단위로 하게 된다면 많은 시간이 소요된다.

그러므로 1AEU 단위로 최적화 하는 것이 아닌 10x10 크기, 5x5 크기와 같이 영역, 즉 구역 별로 공간을 크게 만들어 빠른 시간 내에 최적해를 찾아 내는 알고리즘을 구현하였다.

10x10 크기와 5x5 크기로 공간을 만들었을 때 나온 결과는 그림 4, 5와 같다.

```
▼<Floor-1>
   <Block X="0" Y="0">23</Block>
   <Block X="1" Y="0">24</Block>
   <Block X="2" Y="0">14</Block>
   <Block X="3" Y="0">1</Block>
   <Block X="0" Y="1">21</Block>
   <Block X="1" Y="1">21</Block>
   <Block X="2" Y="1">20</Block>
   <Block X="3" Y="1">5</Block>
   <Block X="0" Y="2">22</Block>
   <Block X="1" Y="2">11</Block>
   <Block X="2" Y="2">7</Block>
   <Block X="3" Y="2">1</Block>
 </Floor-1>
```

그림 4. 1층을 5x5 크기로 변환한 결과

```
▼<Floor-1>
   <Block X="0" Y="0">89</Block>
   <Block X="1" Y="0">40</Block>
   <Block X="0" Y="1">33</Block>
   <Block X="1" Y="1">8</Block>
 </Floor-1>
```

그림 5. 1층을 10x10 크기로 변환한 결과

값이 정확하게 5x5, 10x10의 크기를 가지는 블록(Block)이 나오지 않는 것은 선적할 수 없는 장애물 공간인 Ramp, Beam 등과 같이 충돌 처리에 들어간 공간을 제외하여 결과값을 도출하기 때문이다.

유전자 알고리즘은 교차, 변이, 생존, 우수한 종자를 결과값으로 반환하는 알고리즘이지만 공간 최적화를 하는 과정에서는 교차를 수행하는 것이 복잡하며 목적에 부합하지 않는 결과값이 나올 수 있기에 배제하였다. 그래서 교차 없는 변이와 생존을 통하여 우수한 유전자를 탐색하는 변형된 알고리즘을 구현하였다.

제안한 유전자 알고리즘은 기존과 다른 방식으로 접근하기 때문에 변형된 유전자 알고리즘이며 공간 최적화에 적합한 알고리즘으로 판단된다.

변형된 유전자 알고리즘에서의 변이 알고리즘은 공간의 병합, 공간의 분할 2가지 방법으로 변이가 진행된다. 병합과 분할을 반복하여 공간의 최적화를 찾아가는 이 알고리즘을 Merge and Branch 알고리즘이라고 명명하였다.

병합과 분할에서 병합은 A 공간이 B 공간으로 합쳐진다고 할 때 POD, 도착 지점이 같은 데이터 기준으로 묶는다. 만약에 B 공간이 부족 하다면 더 이상 병합이 불가능하다면 다시 A 공간의 병합하다가 실패한 크기를 저장한다.

분할은 A 공간의 50%를 자른 뒤 B 공간에 이동을 하는 것을 말한다. 분할은 이미 존재했던 공간에 병합하는 것이 아닌 완전히 새로운 공간에 할당한다. 만약에 새로운 공간이 없으며 할당이 불가능하다면 분할을 하였던 A 공간으로 다시 복귀한다.

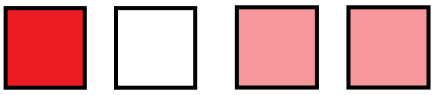

그림 6. 분할하는 과정

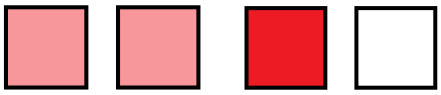

그림 7. 병합하는 과정

그림 6과 7에 나타나 있는 것처럼, 색이 옅은 이유는 적은 양을 차지하고 있다는 모습이며, 짙은 빨간색은 많은 양의 공간을 차지하고 있다는 의미이다.

추가적으로 여러 가지의 알고리즘이 필요로 하겠지만 핵심적인 Merge and Branch 알고리즘을 통하여 적합도 함수값을 계산하여 최적화를 진행하게 된다.

Ⅳ. 결과

차량 배치 공간을 최적화하기 위한 유전자 알고리즘에서 적합도 함수는 사용된 블록의 수에 따라서 값이 변화가 생기도록 하였으며, ExitFloor는 선적한 화물이 바깥으로 나가는 층을 의미하며, $Block_N$의 Floor의 층을 뺀 값의 제곱하여 POD가 먼 화물이 가까이에, ExitFloor에 가까이에 있다면 매우 높은 가중치를 주었으며, 사용되지 않은 블록이 많으면 500이라는 추가 가중치를 부여하였다. 가중치는 낮으면 낮을수록 좋은 유전자를 의미한다. 이 논문에서 사용한 적합도 함수는 다음과 같다.

$$Fitness = \sum_{N=0}^{useBlockCount} (Block_N - ExitFloorvert)^2 + (unUseBlockCount * 500)$$

즉, Merge and Branch 알고리즘과 거리가 가까울수록 좋은 유전자를 의미하는 적합도 함수를 통하여 최적화된 차량 선적 데이터를 찾아내는 알고리즘을 구현하였다. 구현된 알고리즘에 의한 최적화 과정의 1세대 결과와 1,000세대 결과는 그림 8과 같다.

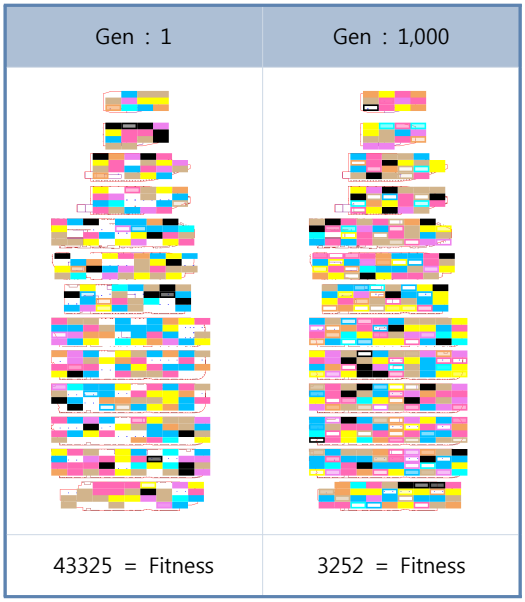

그림 8. 차량 배치 최적화 진화 과정

Ⅴ. 결 론

Merge and Branch 알고리즘과 유전자 알고리즘을 활용하여 차량 선적을 최적화하는 알고리즘을 구현하고 실험 결과를 제시하였다. 최적화를 위한 진화 과정을 통하여 최선의 적합도 값으로 바뀌는 변화를 확인할 수 있었다.

향후 적합도 함수를 더욱 개선시켜 제약 조건, 상태 조건과 같이 여러 가지의 상태와 필요한 조건을 추가적으로 구현한다면 더욱 우수한 결과를 나타낼 수 있을 것으로 예상한다. 또한, Merge and Branch 알고리즘에서 공간 위치를 강제적으로 교환하는 알고리즘 등을 추가하는 방식을 통해 부족한 공간은 그 공간만큼 분할하도록 개선한다면 더욱 빠르게 우수한 유전자를 찾아갈 수 있을 것으로 예상한다.

References

[1] S. K. Kim, et al., “Integrated Method for Optimal Layout of Offshore Plant Topside,” Proc*The Proceedings of Korean CAD/CAM Winter Conference*, pp. 287-291, 2015.

[2] XML Tutorial, w3schools.com. [Internet]. Available : www.w3schools.com/xml/.

[3] Shipping Terms, Carry Cargo International. [Internet]. Available : www.carrycargo.com/shipping-terms/.

[4] J. H. Cha, Y. W. Woo and I. Lee, “An Effective Method for Generating Images Using Genetic Algorithm,” *Journal of the Korea Institute of Information and Communication Engineering*, vol. 23, No. 8, pp. 896-902, Aug. 2019.