6Y-04

隊列走行可能な端末交通システムにおける車両の配送方式

柏木　直樹†　　長谷部　浩二‡

筑波大学情報学群情報科学類†　　筑波大学システム情報系‡

1.　背景と目的

近年、鉄道などの基幹的な交通網と旅客の出発地や目的地とを結ぶ端末交通システムが各地で導入されている。こうした交通システムでは、交通網が複雑になると旅客に乗り換えの負担を強いるという課題が残されている。これを解決するために、車両を電子的に連結することで複数の車両が隊列走行可能な端末交通システムの開発が提案されている[1]。これにより、車両の隊列を再編成することで、乗り換えなしで目的地まで移動することが可能となる。

文献[1]の交通システムでは、交通網が複数のエリアに分かれている。その際に、各車両はどのエリアへ向かうかを行先票を備えることで示し、行先票は車列の再編成を行う際に更新される。しかし、先行研究では、旅客を効率的に運ぶための行先票の割り振り方は検討されていない。

本研究では、車列の再編成をする際に、各エリアの局所的な需要をもとに行先票を割り振ることによって、旅客の待ち時間を削減する配送方式を提案する。また、シミュレーションを用いて提案手法を評価し、その結果が改善されなかった原因を示す。

2.　交通システムの概要

本論文の提案手法を示す前に、文献[1]の交通システムの概要について説明する。複数のエリアが存在し、エリア内には停留所がある。特に、複数エリアにまたがって存在する停留所のことを結節点と呼ぶ。車両は、先導車と後続車に分類される。先導車は有人で後続の車両を牽引しながら走行でき、後続車は無人で先導車に追随して走行する。また、先導車の運行経路は、ある 1 つのエリアを決まった方向に周回するものである。結節点では車列の再編成と行先票の更新をし、そのために切り離した後続車をためておく車両プールを備えている。

再編成のアルゴリズム reshuffle()と、それに用いる変数および関数の定義を以下の表 1 に示す。このアルゴリズムは、エリアを 1 周した車列が結節点に到着した時に実行される。その車列と結節点の車両プールおよび乗車待ちの旅客のリストをもとに、行先票を更新した新たな車列を生成する。最初に、結節点に到着した車列から後続車が切り離され車両プールに入る。その後、車両プールから先導車と行先票が等しい後続車を連結する。旅客の待ち時間を目的のエリア別に合算した際に最大値をとるエリアをその後続車の行先票で示すエリアとする。

表 1：変数および関数の定義

A	エリアのリスト
I	結節点のリスト
i.area	結節点 i の属しているエリア
i.pool	結節点 i がの車両プールに入っている後続車のリスト
lv.lavel	先導車 lv の行先票が示すエリア
tv.lavel	後続車 tv の行先票が示すエリア
lv.tail	先導車 lv が牽引している後続車のリスト
$S(a_1,a_2)$	エリア a_1 で乗車を待つ、エリア a_2 を目的地とする旅客のリスト
S_List(a,A')	エリアのリスト A'に属する要素 a'における S(a,a')のリスト
sort(S)	旅客のリスト S を運行ルート上で近い停留所にいる順（同一停留所の場合は停留所に現れた順）に並び替えたリストを返す関数
s.wait	旅客 s の出発地における待ち時間
s.area	旅客 s の行先エリア
MaxPas	各車両の最大乗車人数

車列の再編成アルゴリズム 1

```
reshuffle(lv, i, S_List(i.area, A))
  for each tv of lv.tail
    i.pool.append(tv);   lv.tail.remove(tv);
  for each tv of i.pool
    if tv.lavel == lv.lavel then
      if tv.lavel == i.area then
        tv.lavel = new_lavel(tv, i, S_List(i.area, A));
      lv.tail.append(tv);   i.pool.remove(tv);

new_lavel(tv, i, S_List(i.area, A))
  lavel = argmax_{a∈A}(Σ_{s∈S(i.area,a)} s.wait);
  S(i.area, lavel) = sort( S(i.area, lavel) );
  delete S(i.area, lavel)[0:MaxPas];
  return lavel;
```

3.　再編成アルゴリズム 1 の問題点

図 1 に本研究で例として扱う交通ネットワークを示す。エリア A_1 ,A_2 ,A_3 ,A_4 があり、上記の再編成アルゴリズム 1 を用いる。それについて Python を用いて表 2 の条件でシミュレーションを行い、旅客の待ち時間を比較した。その結果について、後続車が 30 台の場合を示す。旅客の待ち時間の平均値は

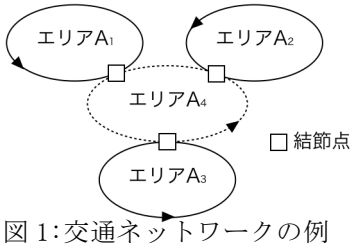

図 1：交通ネットワークの例

A Routing Technique for Last-Mile Transportation Systems with Platooning Vehicles

†Naoki Kashiwagi, College of Information Science, University of Tsukuba.

‡Koji Hasebe, Faculty of Engineering, Information and Systems, University of Tsukuba.

表 2:実行条件

エリア A_1, A_2, A_3 を走る先導車	9 台（各エリア 3 台）
エリア A_i を走る先導車	6 台
後続車	9 台から 50 台（1 台づつ増やす）
旅客の発生確率	各停留所に 0.1 人/分
旅客の移動先	自身の発生場所以外の停留所に対し一様分布
シミュレーションの実行時間	16 時間（午前 6 時から午後 10 時を想定）
最大乗車人数	8 人/台
エリア内の停留所	各 9 ヶ所
停留所間の移動時間	2 分
結節点間の移動時間	5 分

8.5 分、待ち時間の最大値は 31 分となった。待ち時間の最大値が長くなる原因として以下の場合が考えられる。

Case1.　運行経路の後半部分の停留所に来るまでにすでに満車になっている。

Case2.　あるエリア a_1 を目的地とする旅客が長時間待っているが、別のエリア a_2 を目的地とする旅客がそれと比べて多数いると、車両の再編成時に a_2 の行先票のみが割り振られる。

4.　アルゴリズム 1 に対する改善策

以上の 2 つの問題点に対して以下の 2 つの改善を行った。

Case1 への改善策（再編成アルゴリズム 2）

運行経路の前半部分で待っている旅客の集合を S_F、後半部分で待っている旅客の集合を S_L とすると、$\sum_{s\in S_L} s.wait > 2\sum_{s\in S_F} s.wait$ が車列の再編成時に成り立つ場合に、再編成後の車列は後半部分の停留所でのみ乗車を許して運行する。ただし、同一エリア内にこのルールが適用される車列は 1 組のみとする。成り立たない場合はすべての停留所で乗車できるものとする。

Case2 への改善策（再編成アルゴリズム 3）

ある基準となる時間 k よりも停留所での待ち時間の長い旅客がいた場合に、再編成において連結される後続車のうち、少なくとも 1 台はその旅客の行先エリアと同じエリアを示す行先票のバスが連結されるようにする。よって、車列の再編成のアルゴリズム 1 の new_lavel()を以下のように new_lavel2()に変更した。アルゴリズム内の flag は 1 単位時間ごとに True が代入されるブール値である。

車列の再編成アルゴリズム 3 （new_lavel2()のみ）

```
new_lavel2(tv, i, S_List(i.area, A))
 if max_{a∈A}(max_{s∈S(i.area,a)} s.wait) > k & flag then
  tv.lavel = argmax_{a∈A}(max_{s∈S(i.area,a)} s.wait);
  flag = False;
 else
  lavel = argmax_{a∈A}(∑_{s∈S(i.area,a)} s.wait);
  S(i.area, lavel) = sort( S(i.area, lavel) );
  delete S(i.area, lavel)[0:MaxPas];
  return lavel;
```

5.　改善策に対する評価

改善策を、改善前と同じ条件でシミュレーションした。基準となる時間は k=18 分（エリア 1 週分）とした。

その結果、再編成アルゴリズム 1～3 を実行した際の待ち時間の最大値のグラフは以下のようになった。

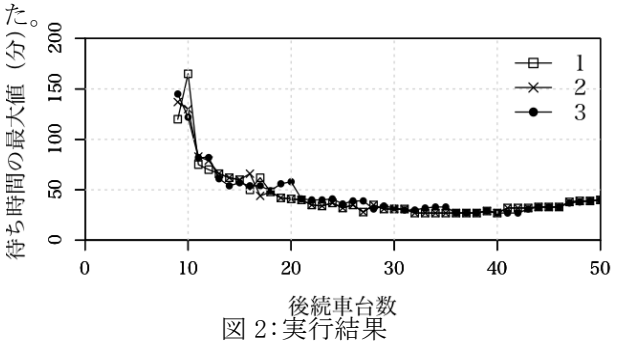

図 2:実行結果

後続車の台数が 30 台の場合において再編成アルゴリズム 1～3 を比較した。その結果、待ち時間の平均値はいずれも約 8 分となった。また、待ち時間の最大値は、再編成アルゴリズム 2 で 32 分、再編成アルゴリズム 3 で 31 分となり、いずれの改善策も効果が得られなかった。

改善が見られなかった原因として、再編成アルゴリズム 2 では、運行経路の前半部分の後半にある停留所と後半部分の後半にある停留所の 2 ヶ所で待ち時間が長くなっていた。

また、再編成アルゴリズム 3 では、基準となる時間 k の値以上に旅客の待ち時間が増大した際に、その旅客が行きたいエリアと同じエリアを示す行先票のバスが連続して出ていたために他の旅客の待ち時間が増大した。

6.　今後の課題

隊列走行可能な端末交通システムにおいて、再編成アルゴリズム 1 を用いて車列の再編成と行先票の更新を行ったところ、旅客の待ち時間の削減をするために解決しなければならない問題点が見つかった。しかし、再編成アルゴリズム 2 では待ち時間の長くなる場所が変わっただけであった。再編成アルゴリズム 3 では少数の待ち時間の長い旅客を重く見過ぎた。したがって、これらの問題点を解決し旅客の待ち時間を削減することはできなかった。

今後は、少数の旅客に対して必要以上の車両が向かわないように行先票を割り振る車両の配送方式を提案し、旅客の待ち時間の最小化を目指したい。

7.　参考文献

[1] K. Hasebe, K. Kato, H. Abe, R. Akiya, M. Kawamoto, Traffic Management for Last-Mile Public Transportation Systems Using Autonomous Vehicles, *3rd IEEE International Smart Cities Conference (ISC2 2017),* 8 pages, 2017.