GTFS と OpenStreetMap を利用した 道路渋滞シミュレーション

白浜 勝太[1]　伊藤 正哉[1]　渕 崇洋[2]　上善 恒雄[1]

[1] 非会員 大阪電気通信大学大学院 (〒 575-0063 大阪府四條畷市清滝 1130-70)
[2] 非会員 大阪電気通信大学 (〒 575-0063 大阪府四條畷市清滝 1130-70)
{*dt16a001, mw19a001, hw16a167*}*@oecu.jp, jozen@osakac.ac.jp*

近年，バスロケーションシステムの普及がめざましく，様々な都市においてバスの到着予告や現在位置が停留所で表示され，インターネットを介しても得られるようになってきた．しかしバス利用者にとって，事前に把握できるバス情報は当てにできなく，臨機応変な対応が必要で，自らのバス移動を含む行動予定の計画が難しい．本来であれば鉄道と同等の精確な時刻表が求められる．また，社会へのデジタル情報配信の進展により，状況に対応した動的時刻表も視野に入ればバス事業者の最適な配車計画に結びつき，経営の効率化にも寄与できる．本研究では，渋滞の可能性を把握することで精確な時刻表設定を支援するために，バス運行シミュレーションシステムの構築を試みている．このシミュレーションシステムは，バスの GTFS データと OpenStreetMap の道路情報を利用し，数学的渋滞モデルを適用する仕組みをベースにしている．

Key Words: シミュレーション，バス，*GTFS*，*OpenStreetMap*

1. はじめに

近年，GPS（Global Positioning System）測位を用いたバスロケーションシステムが増加してきた．それにより，ユーザは，外出先でもスマートフォンからインターネットを介して，簡単にバスの現在位置にアクセスできるようになった．しかし，バスは電車とは違い，不確定要素が多く交通状況や天候状況によってバス到着時刻が左右される．そのため，渋滞時の精確なバス到着時刻を期待することは難しい．現在多くのバスロケーションシステムは，何時にバスが到着するのか予測するのは難しい．様々な要因でバスが遅延しても，精確なバス到着時刻を把握できれば，ユーザは行動決定しやすい．そこで，筆者らは精確なバス到着時刻をユーザに提供することを目標に，GTFS（General Transit Feed Specification）のデータがあれば，どのバス交通事業者の路線であっても扱える簡易的な道路渋滞シミュレーションを構築した．本研究では，バスの運行経路の情報を取得するために，バス運行における様々な公共交通機関情報の共通フォーマットである GTFS の運行経路のデータと，運行経路上の道路情報を取得するために，道路上の様々なデータを含む地理情報データベースの OSM（Open Street Map）のデータの 2 つのデータを扱う．本システムでは，この 2 つデータを組み合わせたデータと，文献[1] の最適速度数理モデル (Optimal Velocity Model) を利用し，走行車の速度の計算を行った．

2. 関連研究

文献[2] では，バスロケーションシステムのデータを用いて，早発を防ぎ遅延が少ないダイヤを自動生成する研究を行っている．バスロケーションシステムからバス停の通過時刻を収集することで遅延の少ないダイヤを AI が自動生成するシステムを構築しており，AI が生成した改正ダイヤと改正前ダイヤを比較すると，最大遅延時間が半減したと報告している．しかし，渋滞有無の判断はダイヤ担当者がしており，渋滞によるダイヤの調整は自動化されていない．

文献[3] では，バスの遅延時間に対して道路データとバス運行データがどの程度影響しているのかを重回帰分析を用いて遅延要因分析しており，バス遅延要因のひとつが信号数と報告している．

文献[4] では，バス到着時刻予測として，バスロケーションシステムが収集・蓄積したデータを用い，重回帰モデルからルート毎の傾向を算出し，それを初期入力としてカルマンフィルタを利用し予測精度を更新している．しかし，予測精度の更新が最適に行われない問題がある．

3. シミュレーションに必要なオープンデータ

渋滞シミュレーションを構築する上で，必要としたデータは下記のとおりである．

- バス運行路線
- バス運行路線の道路情報
- 各車両の速度

これらのデータの取得を実現するにあたって，GTFS や OSM といったオープンデータから取得した．

(1) GTFS

GTFS とは，国土交通省が推奨している，公共交通機関の時刻表と地理的情報に関するオープンフォーマットである．GTFS の feeds により、公共交通機関は交通データを公開しており，そのデータを相互運用可能な方法で使用することができる．文献[5] によると，コミュニティバスや中小バス事業者が徐々にバス情報を GTFS として公開している．

GTFS のデータは下記の様な CSV ファイルで構成されている．

- agency - 交通機関基礎情報
- stops - 停車地，バス停
- routes - 交通機関のルート
- trips - 各ルートを構成する旅程
- stop_times - 停車地への到着時刻と出発時刻
- calendar - 曜日指定の運行日
- calendar_dates - 日付指定の運行日
- fare_attributes - ルートの運賃情報
- fare_rules - 運賃の適用ルール
- shapes - 車両の移動経路
- frequencies - 運行間隔（時刻表がない場合）
- transfers - 乗り換え補足情報
- pathways - 駅構内のレイアウト
- levels - 駅構内の階層
- feed_info - GTFS フィード自体についての情報

GTFS の shapes ファイルに各バス路線の移動経路の位置データが保存されており，その位置データをつなげることでバス路線を取得できる (図 1).

後述する OSM 道路データとのマッチングのために均等な間隔のバス路線データが必要なため，取得したバス路線のデータを 1m 毎に区切る．

(2) OSM 道路データ

OSM とは無料，かつ全てのユーザが編集可能な世界地図であり，数多くのボランティアによりゼロから作られている．この OSM のデータを 1 つにまとめた Planet.osm というファイルがあり，地図を構成するノード，ウェイ，リレーションのデータが入っている．この

図–1 shapes のデータ

Planet.osm は新しいバージョンを毎週リリースしており，オープンデータのため，簡単にデータを取得することができる．その Planet.osm のデータを PostgreSQL というデータベースに格納することで，PostGIS でのデータ取得が可能になる．PostGIS は PostgreSQL データベースで地理空間情報を扱うための拡張機能のことである．これを用いることで必要な区間だけの OSM 道路データを取得することが可能になる．Planet.osm のデータを PostgreSQL に格納すると下記の様なテーブルを取得できる．

- planet_osm_line
- planet_osm_node
- planet_osm_point
- planet_osm_polygon
- planet_osm_rels
- planet_osm_roads
- planet_osm_ways

この中から車道，信号機，交差点，制限速度，車線数，トンネルの出入口などの OSM 道路データを取得したい．planet_osm_line には様々な道の情報が格納されており，このデータの highway 属性から車道だけを指定し，制限速度，車線数，トンネル，交差点のデータを取得することができる．planet_osm_point には様々な点の情報が格納されており，このデータの highway 属性から信号機，junction 属性から交差点のデータを取得することができる．

GTFS のデータと OSM 道路データを組み合わせるため，R-tree[6] に利用した．R-tree とは，木構造のデータ構造であり，位置座標データにインデックス付けをする．OSM 道路データを R-tree を用いて，位置データのインデックス付けを行い，1m 毎に区切ったバス路線の点データの近傍点を検索し，点データと OSM 道路データの紐付けを行った．

4. 道路渋滞シミュレーション

(1) シミュレーションに必要な要素

まず，走行車の速度は，最適速度数理モデル[1] を利用する．

$$V(\Delta x) = tanh(\Delta x - 2) + tanh2$$

しかし，この数理モデルのままだと，前方の走行車の速度に合わせた速度になるため，OSM 道路データに合わせたモデルに調整する必要がある．OSM 道路データにより，走行車の速度が変動するので，ms というパラメータを定義する．

$$V(\Delta x) = (tanh(\Delta x - 2) + tanh2) \cdot ms$$

次に，様々な道路状況による走行車の速度の変化を考える必要がある．筆者らは，以下の点で速度が変化すると推測した．

a) 制限速度

OSM 道路データの tags 属性から maxspeed の値を取得できた場合，これを ms に設定する．

$$ms = maxspeed$$

取得できなかった場合は，法定速度 lv を設定する．

$$ms = lv$$

b) 交差点

交差点の箇所は，OSM 道路データの highway 属性，junction 属性のデータから取得することが可能である．走行車は交差点の規模によって交通状況が変動する．現在走行している道路が主要な道路なのか，交差点で右折・左折時に走行する道路が主要な道路なのかにより，走行車の動きに違いがあると推測する．直進する可能性を sp，右折・左折する可能性をそれぞれ rlp と定義する．主道路を走行時は，sp の値は rlp より高く，従道路を走行時は，sp の値は rlp より低いと考える．直進する場合は速度が変わらないが，右折・左折時には曲がるために速度が変わるため，ms の値を変更する必要がある．また，右折時と左折時に出す速度は別々だと推測し，右折時に出す速度を rs，左折時に出す速度を ls と定義する．

OSM 道路データは，文献[7] から，交差点の情報が過多である点が報告されていることは留意する必要がある．

c) 信号機

信号の箇所は，OSM 道路データの highway 属性から traffic_signals の値を探すことで判別する．文献[8] によれば，信号機の色の変わるタイミングは，サイクル，スプリット，オフセットの 3 種類ある．一般的な車両用信号機の場合，青色，黄色，赤色という順に色が一巡する時間をサイクル（周期）と呼ぶ．1 サイクル中の青色の時間を bt，黄色の時間を yt，赤色の時間を rt と定義する．1 サイクル中の割り当てる時間配分をスプリットと呼ぶ．スプリットは交通量の多い道路側を主道路とし，時間配分を多く設定する必要がある．そのため，OSM 道路データから周期を求める上で，交差点での交通量の多い主道路側と，多くない従道路側の判別を行う必要がある．この判定は，OSM 道路データから lanes の値の差，もしくは道路名のデータから可能である．主道路側のスプリットを ml，従道路側のスプリットを sl と定義する．各交差点を円滑に通過できるように，隣接する交差点の青信号開始時間をずらす必要がある．このずれをオフセットと呼ぶ．そのため，走行する道路が主道路の場合，連続した交差点を検知する必要がある．連続した交差点は，planet_osm_line のデータから主道路の線データを取得し，planet_osm_point から信号機のデータを組み合わせることで連続しているかどうか判別ができる．オフセットの値を o と定義する．信号機の色が赤色・黄色のときは，ms の値を 0km/h に設定する．

d) カーブ

カーブの箇所は，1m ずつ区切った点の位置情報から計算する．R 値を求めることで，どの程度の大きさのカーブなのかを求めることができる．カーブの大きさによって出す速度を cv と定義する．また，a) で設定した ms より cv が高い場合は．ms の値を優先する．

e) トンネル

トンネルの出入口の箇所は，OSM 道路データの tunnel 属性の値から取得することが可能である．走行車が一定の速度を超えているときのみ，減速するように設定する．この一定の速度を cv と設定し，これを超えた時に減速する値を dv と定義する．cv は a) の ms によって変動する．

$$V(\Delta x) = (tanh(\Delta x - 2) + tanh2) \cdot ms - dv$$

f) 坂道

坂道の箇所は，OSM 道路データの tags 属性の ele の値から高度を取得することが可能である．なので，バス路線上の各点の高度の差を計算することで取得することが可能である．各点の高度の値から縦断勾配の値を求めて，この値によって走行車の速度を調整する．縦断勾配が 1 %毎に速度が上下すると推測する．縦断勾配によって上下する速度を sv と定義し，縦断勾配の値を s と定義する．

$$V(\Delta x) = (tanh(\Delta x - 2) + tanh2) \cdot ms - s \cdot sv$$

ただ，OSM 道路データの高度データ設定されてない場合が多く，SRMT（Shuttle Radar Topography Mission）データなど，他のオープンデータと併用する必要がある．SRMT とはスペースシャトルに積んだレーダを用いて，全世界の立地地形データを作成するプロジェクトである．

g)　車線数

車線数は，OSM 道路データの tags 属性の lanes の値から取得することが可能である．走行車がどの車線を走行するかは，空いている車線，次の交差点で左折できる車線，次の交差点で右折できる車線の判別が必要になる．走行車は交差点通過時に次の交差点で直進，右折，左折のどの動作を行うのか b) で設定した割合で，この時点で決定する．決定した時，次の交差点まで専用車線がある場合は，その車線を走行する．ない場合は，空いている車線を走行すると定義する．

OSM 道路データは，文献[7] から，車線数は交差点を境界にして極端に変化する点が報告されていることを留意する必要がある．

h)　走行車数

季節，月，日，曜日，1 日の朝昼夜等の時間情報によって走行車数を検討する必要がある．この値は，オープンデータである GTFS のデータや OSM 道路データから求めるのは現時点では難しい．走行車数は N と定義し，適当な値を設定する．

i)　バス路線距離

バス路線距離は，GTFS のバス路線データから，求めることができる．バス路線距離は P と定義する．

(2)　現在の実装

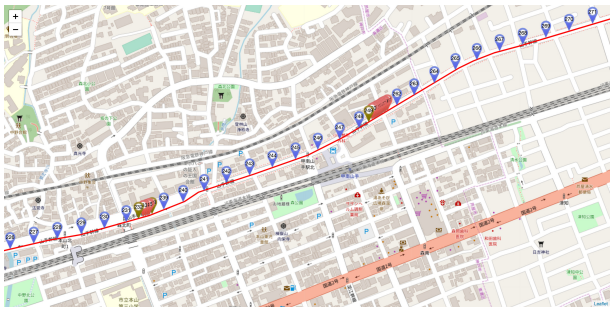
今回は，a) と c) の一部を渋滞シミュレーションに実装した．lv は 60km/h と設定し，ms は，制限速度を取得できた場合は，その値を設定し，それ以外は lv を設定する．すべての信号機は，1 サイクル 100 秒と設定し，bt を 57 秒，yt を 3 秒，rt を 40 秒と設定した．P は 12488m，N は 300 台と設定した．

渋滞シミュレーションは Web 上で表示するように構築した（図 2 ）．JavaScript を用いて，各走行車の速度計算を行い，位置情報の表示を行った．地図上のマーカが走行車を表しており，マーカ上部についている数字は走行車の ID を表す．マーカの色は速度を表しており，下記のように設定している．

- 赤 - 0km/h 以上，20km/h 未満
- 黄 - 20km/h 以上，40km/h 未満
- 緑 - 40km/h 以上，60km/h 未満
- 青 - 60km/h 以上

5.　まとめ

筆者らは，最適速度数理モデルを利用して，各交通状態での速度計算の定義を行い，GTFS データと OSM 道路データがあれば渋滞シミュレーションが可能になるシステムを構築した．この渋滞シミュレーションはオープンデータを用いているため，容易に Web 上で確認することができる．



図–2 シミュレーション

6.　今後の予定

今後，最終目標であるユーザが精確なバス到着時刻を取得を可能にするために．今回定義を行ったが，実装できてない値をシミュレーションに実装し，各速度計算の検証を行う必要がある．できるだけオープンデータを用いて，走行車数と高度の取得ができるか検討する．バスロケーションシステムの収集・蓄積データから走行車両数の推測，時期・時間帯からの渋滞頻度，渋滞箇所等を求め，実データと渋滞シミュレーションのデータを比較し，評価する．その結果から今回定義した変数，または速度の計算方法を調整し，渋滞シミュレーションの精度の向上を図る．

参考文献

1) M.Bando，K.Hasebe，A.Nakayama，A.Shibata and Y.Sugiyama，Structure Stability of Congestion in Traflic Dynamics，日本応用数理学会，1994 年
2) 太田恒平，森慶太，平本清志，前川雄祐，伊藤昌毅，バスロケデータを基にした路線バスの遅延対策ダイヤ改正〜両備グループにおけるオープンイノベーションの実践〜，第 57 回土木計画学研究発表会，2018 年
3) 藤原由美恵，花田智，白石陽，道路データとバス運行データを用いたバス遅延要因分析，情報処理学会第 78 回全国大会，2016 年
4) 今井瞳，廣井慧，河口信夫，路線バスの運行データ分析に基づく到着時刻予測と精度解析，DICOMO，2016 年
5) 伊藤昌毅，瀬崎薫，日本における公共交通オープンデータの現状と展望，第 55 回土木計画学研究発表会，2016 年
6) Antonin Guttman，R-Trees - A Dynamic Index Structure for Spatial Searching，ACM SIGMOD，1984 年
7) 芦田拓人，藤井秀樹，内田英明，吉村忍，オープンデータを用いた交通流シミュレーションにおける道路ネットワーク適正化，人工知能学会全国大会（第 30 回），2016 年
8) 公益財団法人日本交通管理技術協会，交通整理 ABC，https://www.tmt.or.jp/research/img/signal/s-02.html，2019 年

(2019. 10. 04 受付)

TRAFFIC CONGESTION SIMULATION USING GTFS AND OPENSTREETMAP

Shota SHIRAHAMA, Masaya ITO, Takahiro FUCHI and Tsuneo JOZEN

In recent years, bus location systems have become very popular, and bus arrival notices and current locations are displayed at bus stops in various cities, and are also available via the Internet. However, for bus users, bus information that can be grasped in advance cannot be relied on, and it is necessary to respond flexibly, so it is difficult to plan an action plan that includes own bus movement. Originally, an accurate timetable equivalent to the railway is required. In addition, with the progress of digital information distribution to society, if a dynamic timetable corresponding to the situation is taken into consideration, it will lead to an optimal dispatch plan for bus operators and contribute to management efficiency. In this study, we are trying to construct a bus operation simulation system to understand the possibility of traffic jams and support accurate timetable setting. This simulation system is based on a mechanism that uses GTFS data of buses and road information of OpenStreetMap and applies a mathematical traffic jam model.