ライドシェア普及時のための乗客旅行時間を最小化する信号制御手法

飯塚　叶恵[*1]　　瀬尾　亨[*1]
東京工業大学　環境・社会理工学院[*1]

適切な交通信号制御は混雑した都市道路網の効率化に大きな役割を果たしている．既存の信号制御では，車両台数に着目した旅行時間や遅れ時間の最小化を目的とする例が多い．しかし，近年ライドシェアの普及が進んでおり，車ではなく人の利便性の観点から乗客数の多いライドシェア車両を優先する制御も考えられる．本論文では，各車両の乗車人数を考慮し人々の総旅行時間を最小化する手法を動的計画法に基づき提案する．このとき，人々の総旅行時間だけを考慮した場合に問題になる，車両待ち行列のスピルバックや特定個人の待ち時間の増大を防ぐような手法を組み込んだ方法とする．

Traffic signal control that minimizes the passenger travel time for ride-sharing service

Kanae Iizuka[*1]　　Toru Seo[*1]
Tokyo Institute of Technology, School of Environmental and Society[*1]

In a congested urban road network, it is important to minimize the passenger travel time and the delay through appropriate traffic signal controls. Previous studies have mainly focused on the vehicle travel time or queue lengths around the signal. However, as ridesharing has become popular in recent years, it is also effective to prioritize ridesharing vehicles in terms of the convenience of people rather than cars. In this paper, we propose a control method based on dynamic programming that minimizes the passenger travel time and prevents vehicle spillback by taking into account the number of passengers of each vehicle.

Keyword: ridesharing, adaptive signal control, passenger-based control

1. 序章

適切な交通信号制御は混雑した都市道路網の効率化に大きな役割を果たしている．既存の信号制御では，車両台数に着目した旅行時間や遅れ時間の最小化を目的とする例が多い．しかし，今後ライドシェア(RS)が普及した場合には，乗客数・乗車人数に着目した乗客ベースの制御により人が経験する渋滞の直接的なコストを減らすことが有効と考えられる．長期的な視点では，乗客ベースの制御がモーダルシフトを引き起こし，公共交通やライドシェアの利用の促進が期待できる．

しかし，乗客ベースの信号制御にはいくつかの根本的課題がある．まず，これは spillback を助長する恐れがある．例えば，RS の割合が高く，乗客ベース

の需要が大きいレーンと，一人乗りの車(SD)が多く乗客ベースの需要が小さいレーンがあるとき，素朴な乗客ベース制御を適用すると前者が優先されるため，後者のレーンの列が延伸し spillback が生じる可能性がある．車両ベースで旅行時間を最小化する際は列の長いレーンが優先され，spillback は発生しにくいため，これは乗客ベース特有の課題と言える

また，特定の個人の待ち時間が極端に長くなる恐れもある．例えば，一つのレーンの需要が他に比べ極端に少ない場合，他のレーンが優先され続け，特定の車両が長く待たされ，著しく不公平になる可能性がある．この現象は車両ベースの制御でも生じうるが，乗客ベースにすることでより各 phase の評価のばらつきが大きくなり起きやすくなる場面もある．

本研究の目的は，RS の普及を念頭に，孤立した信号交差点を対象とし，各車両の乗車人数の情報を用いて一定期間の乗客旅行時間を効率よく減少させる交通信号制御の提案である．Sen and Head [1]によって提案された動的計画法による手法 COP (Controlled Optimization of Phases)を拡張し, 乗客ベースの制御にすることで顕在化しうる２つの課題を，ペナルティ関数法により制約式を目的関数に組み込む新たな方法を開発し解決する．

乗客数を考慮した制御について, Yu ら[2]は車とバスを人数で重みづけし遅れ時間を最小化する数学的アルゴリズムを提案した．Wu ら[3]は人々の遅れ時間を動的計画法により最適化する手法を提案した．また，Spillback を避けるために，列の長さが閾値を超えた際は目的関数を車両ベースの junction throughput に変更している. 特定の車両を優先する制御は，バスや緊急車両を対象にしたものが多く，一般車の人数を直接考慮した研究は知る限り[3]のみである. [3]では, 目的関数を変更することで spillback を防いでいるが，列の長さの制約を動的計画法の目的関数に直接組み込み最適化した研究は見られない．制約を直接組み込むことで，列の延伸を観測する前に，予測される列の延伸を未然に防ぐような制御とすることができる．個人の待ち時間に関しては，最大赤時間を制限する方法もあるが(Van et al. [4])，待機車両の無い場合に非効率が生じてしまう．

2. 既存手法 COP の概要

COP は Sen and Head [1]によって提案された，独立交差点において一定期間の中で目的関数を最適化するような青時間の配分を，動的計画法により求める手法である．これは rolling horizon と併用され，一度の計算で最適化する期間を prediction horizon Tとする．各 stage は phase に対応しており，stage を一つずつ増やしながら最適解を探索する. phase が一巡すると次の stage は一つ目の phase から再び紐づけられる．最終的な stage 数や phase の順番は未知な状態で探索が行われ，stage に使う時間が 0 のとき，その stage に対応する phase は飛ばされることを意味する．COP は，stage を増やしながら探索する forward recursion と，その結果から最適制御を計算する backward recursion から構成される．

ここで，変数を以下のように定義する．s_jは stage jが終わる時刻タイムステップを表し，stage jの青時間x_jに応じて stage jに割り当てられる時間$h_j(x_j)$を用いて式(1)のように表される．

$$s_{j-1} = s_j - h_j(x_j) \tag{1}$$

$h_j(x_j)$は，式(2)のように$x_j > 0$の場合には全赤時間rが足される．

$$h_j(x_j) = \begin{cases} 0, & \text{if } j \geq 2, x_j = 0 \\ x_j + r, & \text{otherwise} \end{cases} \tag{2}$$

これらの関係を表したものが図 2-1 である．各s_jに対して可能な$x_j \in X_j(s_j)$は，最小青時間γと全赤時間rを用いて式(3)のように定義される．

$$X_j(s_j) = \begin{cases} \{0\}, & \text{if } s_j - r < \gamma \\ \{0, \gamma, \gamma+1, \dots, s_j - r\}, & \text{otherwise} \end{cases} \tag{3}$$

また，目的関数は value function $v_j(s_j)$で表され，各 stage jでの performance measure $f_j(s_j, x_j)$の和として式(4)のように定義される．

$$v_j(s_j) = \sum_{j'=1}^{j} f_{j'}(s_{j'}, x_{j'}) \tag{4}$$

つまり，COP は performance measure の総和を最小化するように，将来の交通状況を予測しながら現在の信号現示を決定する手法である．Sen and Head [1]では，performance measure は最大列長，停止回数，車両遅れ時間として定義され，車両ベースの信号制御手法が提案された．

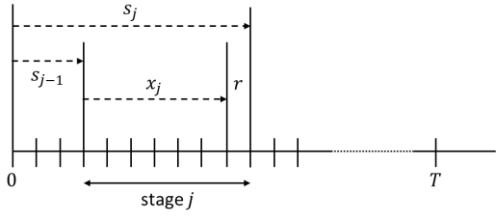

図 2-1. s_jとx_jの関係

2-1 Forward recursion

まずは stage $j=1$，つまり phase を一つのみ考慮する場合から探索する．初期化として，$v_0(s_j)$を定義する．そして$s_j=r$に対して可能なx_jを一つ定め，stage jの間の performance measure$f_j(s_j,x_j)$を計算し，stage $j-1$までの積算値$v_{j-1}(s_{j-1})$と足し合わせることで，stage jまでの value function を求める．これを全ての$x_j \in X_j(s_j)$について計算し，value function を最小にする最適解$x_j^*(s_j)$と対応する$v_j(s_j)$を記録する．s_{j-1}時の車の状況$Q_{l,j-1}(s_{j-1})$と$x_j^*(s_j)$を用いて，$Q_{l,j}(s_j)$を更新する．このとき，$Q_{l,j}(s_j)$は[3]を参考に各車両の予想リンク出発時刻を用いる．これを$s_j=r+1,\dots,T$について繰り返し，全てのs_jを探索し終えたら，次の stage に進むかを判定する．このとき，最低でも全ての phase を探索する必要がある．また，最後の$|P|-1$個の stage の全てのs_jにおいて，$x_j^*(s_j)=0$のとき，新しく stage を足すことにより目的関数を改善できないため，終了とする．これらをまとめると以下のようになる．

Step 1．初期化$v_0(s_j)=0$，$j=1$
Step 2．各$s_j=r,\dots,T$に対して，

$$v_j(s_j)=\min_{x_j}\{f_j(s_j,x_j)+v_{j-1}(s_{j-1})|x_j\in X_j(s_j)\}$$

上記問題の最適解$x_j^*(s_j)$と対応する$v_j(s_j)$を記録し，$Q_{l,j}(s_j)$を更新
Step 3．$j<|P|$ならば $j\leftarrow j+1$とし，step 2 から繰り返す．
全ての$k\le|P|-1$と$1\le t\le T$に関して$x_{j-k}^*(t)=0$ならば終了．
上記以外の場合 $j\leftarrow j+1$とし，step 2 から繰り返す．

2-2 Backward recursion

次に，forward recursion で求めた各j，s_jに対する最適青時間$x_j^*(s_j)$の表をもとに，$s_j=T$での最適な信号制御を取り出す．まず，探索した最後の stage を Jとし，stage $j=J-(|P|-1), s_j^*=T$での$x_j^*(s_j^*)$を読み取り，stage jでの青時間とする．このとき対応するs_{j-1}^*は，$s_{j-1}^*=s_j^*-h_j\left(x_j^*(s_j^*)\right)$により求まる．次に，これを用いて$j=J-(|P|-2)$の時の$x_j^*(s_j^*)$を読み取り，その stage での青時間とする．これを$j=1$まで繰り返すことで，各 stage の最適青時間を計算する．このプロセスは以下のように表せる．

Step 1．$s_{J-(|P|-1)}^*=T$
Step 2．各$j=J-(|P|-1),\ J-(|P|-2),\dots,1$に対して，Forward recursion で計算された表から$x_j^*(s_j^*)$を読み取る．$j>1$ならば，$s_{j-1}^*=s_j^*-h_j\left(x_j^*(s_j^*)\right)$とし，これを繰り返す．

3. 提案手法

提案手法では，乗客の旅行時間の最小化を目的とした制御を行う．これは，COP の performance measure の定義をその目的に適したものとすれば実現できる（3-1 節）．そのうえで，乗客旅行時間最小化に伴う固有の問題を解決するための方法を組み込む（3-2, 3-3 節）．これらは，図 3-1 に示すフローチャートの赤枠部分で行われる．

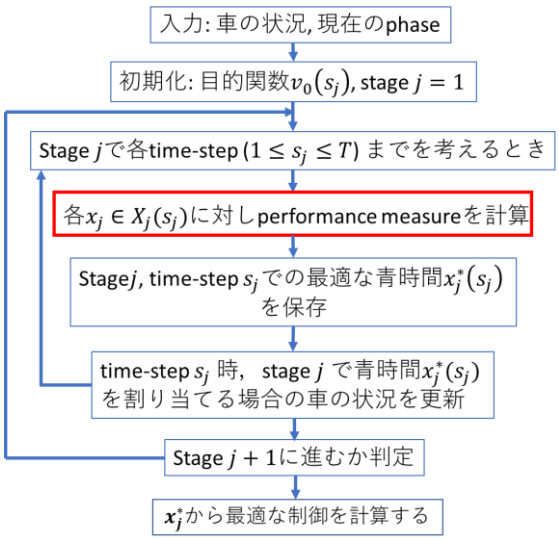

図 3-1．提案手法のフローチャート

3-1 人々の旅行時間

各 stage jの評価$f_j(s_j,x_j)$は，stage jの期間における，交差点への各流入リンクl上の人々の旅行時間$TTS_{l,j}(s_j,x_j)$の総和として式(5)と定義する．

$$f_j(s_j,x_j)=\sum_l TTS_{l,j}(s_j,x_j) \tag{5}$$

3-2 列長の制限

冒頭にも述べたように，乗客ベースで制御することで，列が延伸し spillback が起きる可能性がある．本研究では，ペナルティ関数法を用いてこれを制限する．ペナルティ関数法とは，制約式を目的関数にペナルティとして組み込むことで，制約付き最適化問題を制約なし問題にする方法の一つである．ペナルティは，制約式を満たしている間は与えられず，制約式を満たさなくなった場合にその差分に応じて

与えられる．そのため，制約式を多少満たしていなくとも解が求まるのが特徴であり，需要が大きく列の延伸を避けられない場合にも適用できる．ここで，列長制限を組み込んだ目的関数は excessive queue area $z_{l,j}(s_j,x_j)$を用いて式(6)のように表される．

$$f_j(s_j,x_j)=\sum_l\left(TTS_{l,j}(s_j,x_j)+\alpha z_{l,j}(s_j,x_j)\right) \quad (6)$$

パラメータαは大きいほどより強く列長の制限が働く．Excessive queue area とは，閾値を超えた分の列長を時間方向に足し合わせたもので，車両軌跡図上では，図 3-2 のように，リンク長に対する列長の割合で表される閾値λ_{queue}，列最後尾，s_{j-1}，s_jに挟まれた部分の面積で表される．実装時は，stage $j-1$が終わるs_{j-1}での列の最後尾の位置，stage jの間に列に加わる最後の車両の，列に加わる位置と時刻を予測することで求める．stage jで対象リンクに対応する phase が青の時は，これらに加え，最後尾が動き始め列が解消する時刻も予測し考慮する．

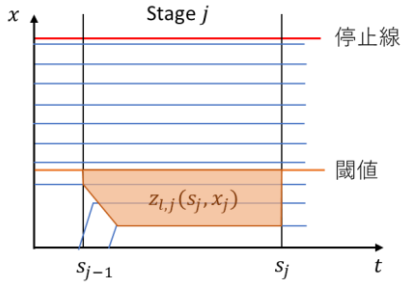

図 3-2. Excessive queue area

Excessive queue area では，時間方向に足し合わせることで，stage 数に左右されない列長の評価が可能となっている．仮に stage 終了時の瞬間列長をペナルティとして用いた場合，同じ期間，状況の列長の評価であっても stage 数の増加に伴いペナルティも増加するため，結果として stage 数を減らすような制御となってしまう．また，列長に応じたペナルティであることに加え，列長が閾値を超えていた時間にも比例したペナルティを与えられることもメリットとして挙げられる．これにより，列長が閾値を超える時間が短くなるような制御となる．

3-3 待ち時間の制限

生じうる極端に長い個人の待ち時間に関しても，列長同様ペナルティ関数を用いて制限する．目的関数は，リンクの先頭車両の停止線での待ち時間が許容待ち時間λ_{wait}を超過した分の時間$w_{l,j}(s_j,x_j)$を用いて式(7)のように表される．

$$f_j(s_j,x_j)=\sum_l\left(TTS_{l,j}(s_j,x_j)+\alpha z_{l,j}(s_j,x_j)+\beta w_{l,j}(s_j,x_j)\right) \quad (7)$$

βはα同様，大きいほどより強く待ち時間の制限が働く．ここで先頭車両の停止線での待ち時間を考慮する理由は，先頭車両はその列内のどの車両よりも待ち時間が長いためである．車両は複数 stage に渡って停止線で待つ可能性があるため，$w_{l,j}(s_j,x_j)$は前の stage での値を用いて更新される．

4. シミュレーションによる検証

提案した乗客ベース COP を検証するため，Newell の単純追従モデルをベースとした交通シミュレータ UXsim [5,6]と組み合わせてシミュレーションを行った．

4-1 シミュレーション環境

図 4-1 のように 4 方向の独立交差点を一つ考え，4 つの phase が割り当てられるとする．交差点に流入するリンクは各方向から左折直進用のレーンと右折専用レーンの 2 レーンあり，各レーンの長さは 500m である．レーンは独立しており，各車両は交差点通過後の進行方向に合わせてレーンを選び，レーン進入後の車線変更はないものとする．車両は一人乗りの SD と 4 人乗りの RS を考え，ノード 2〜5 から各方向に出発する．シミュレーションでは，UXsim 内で信号が既知の期間に対して車両軌跡を更新し，交差点に流入するリンク上の車の速度・位置・車種と現在の phase の情報は COP に渡される．COP では，与えられた情報から，最適な次の phase とその長さを計算し，UXsim に渡す．

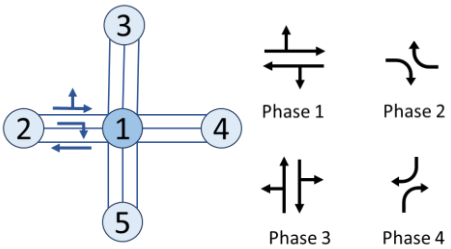

図 4-1. 対象交差点と phase

新しい需要の考慮や予測ミスの誤差の補正を行うため，rolling horizon を適用する．COP の計算結果のうち，最初の phase のみを適用し，次の COP に移る．ただし，最初の phase が 10 秒以上続く場合，情報の更新が長時間行われないことを避けるため，10 秒で次の COP を行う．

4-2 パラメータ設定と需要

車の挙動に関して，自由流速度を 12[m/s]，反応時間を 1 秒，渋滞密度を 0.1[veh/m]とした．また，COP に使われるパラメータは，最小青時間が 10 秒，全赤時間が 1 秒である．ただし，stage 1 に関しては，前の phase からの続きであり，最小青時間は既に経過しているため，最小青時間の制約は適用されない．

３つの各シナリオの需要は表 4-1 に示すとおりである．①は 60 秒間，②は 600 秒間，③は 100 秒間の需要のもとでシミュレーションを行った．シミュレーション時間は，すべての車両が余裕をもってトリップを完了できるよう，①は 200 秒間，②は 1000 秒間，③は 400 秒間とした．

Prediction horizon は，短すぎるとすべての phase の組み合わせを考慮することができない．今回の例では，stage 1 で最低 1 秒，stage 2~4 で各々最低 11 秒からであるため，34 秒以上あればすべての phase を一つの horizon の中で同時に考慮することができる．また，予測可能な車両がすべて捌けられるだけのTが望ましい．裏を返せば，どのような信号制御でも horizon 内に捌けられない車両は旅行時間の低下に貢献できないため，結果に影響を与えない．しかし，Tが大きすぎる場合，計算量が多くなり現実世界においてリアルタイムで結果を反映することが難しくなる．ここで，rolling horizon を併用することで，Tがある程度短くとも全体としておおよその最適性は保たれる．今回 prediction horizon は，①②では 80 秒，③では 120 秒とし，①③では未観測の車両による誤差が小さくなるようにしている．

表 4-1. 各シナリオの需要

出発ノード	到着ノード	シナリオ①		シナリオ②		シナリオ③	
		流率 [veh/s]	RS割合	流率 [veh/s]	RS割合	流率 [veh/s]	RS割合
2	4	0.4	0	0.4	0.4	0.4	0.3
2	5			0.5	0	0.05	0
3	5	0.3	0.6	0.2	0.6	0.2	0.5
4	2	0.35	0	0.3	0.4	0.3	0.3
5	3	0.4	0.6	0.4	0.6	0.2	0.2

4-3 結果

各シナリオの PHT，最大列長，最大待ち時間をまとめたものが表 4-2 である．シナリオ①では，１つのシミュレーションについて結果を掲載している．

①車両ベースと乗客ベース

車両ベースと乗客ベースの COP について，ランダムに RS 車生成のタイミングを変えながら 100 回シミュレーションを行い，SD，RS，全ての車それぞれについて PHT（乗客旅行時間）を比較した．各シミュレーションについて車両ベースの場合の PHT に対する乗客ベースの場合の PHT を計算した結果を図 4-2 に示す．全体として車両ベースの場合に比べ，SD の PHT は悪化しているものの，RS の PHT は 15%ほど改善しており，全体としては約 8%改善している．

表 4-2. 各シナリオの PHT，最大列長，待ち時間

シナリオ		PHT [pas*s]	最大列長 [m]	最大待ち時間 [s]
①	veh base	**10251**	190	41
	pas base	**8987**	200	53
②	none	91344	**490**	462
	$\alpha = 0.5$	89013	**420**	346
	$\alpha = 5$	90477	**420**	352
③	none	22316	250	**106**
	$\beta = 10$	23207	250	**58**
	$\beta = 20$	23346	250	**48**

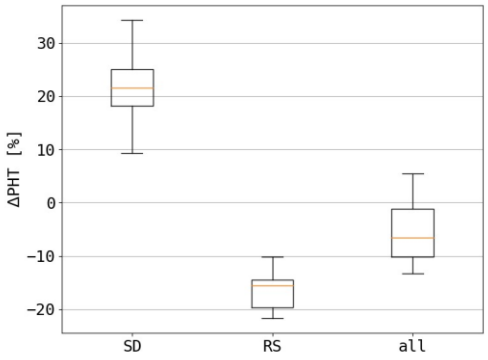

図 4-2. 車両ベース COP に対する乗客ベース COP による制御時の PHT

②列長の比較

spillback が生じるような需要のもとで，$\lambda_{\mathrm{queue}} = 0.7$ と異なるαを用いて列長制限を課すと，交差点周りの時間ごとの最大列長は図 4-3 のようになる．縦軸はリンク長で正規化されている．αが大きいほど列長制限の影響が大きくなり，列の延伸が抑えられていることが分かる．表 4-2 では列長制限を課すと本来増加するはずの PHT が減少しているが，これは，horizon 内に新たに流入する未観測の需要を考慮できずに生じた誤差の影響であり，制限を課すことで未観測の需要にとって都合の良い制御となったためである．今後需要予測を組み込むことで，制限なしの PHT は下がると考えられる．

③待ち時間の比較

需要が少なく待ち時間が極端に長い車両が出てしまう需要下で，$\lambda_{\text{wait}} = 50$と異なる$\beta$を用いて待ち時間の制限を課すと，各車両の交差点での待ち時間の分布は図 4-4 のようになる．何も制限を課さない場合，各車両の待ち時間は最大 200 秒程に達するが，制限のある場合，βが大きくなるにつれ最大待ち時間が 50 秒に近づいている．また，表 4-2 から，待ち時間制限を課すことで PHT が増加していることが分かる．

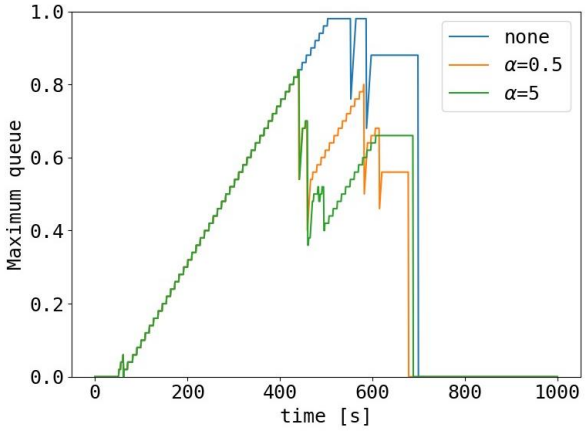

図 4-3. 交差点周りの時間ごとの最大列長

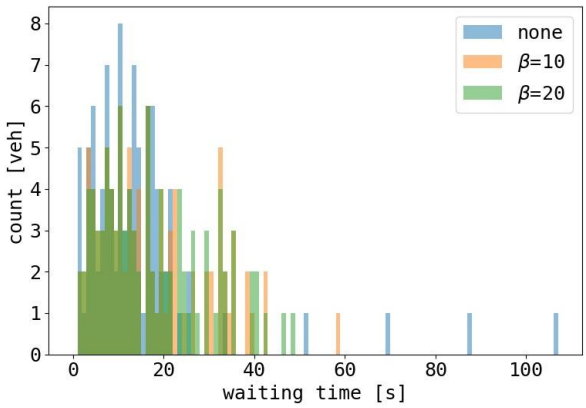

図 4-4. 各車両の交差点での待ち時間分布
（待ち時間が 0 秒の台数は各 18 台）

4-4 考察

COP では，各 stage，各s_jで最適な各x_jを一つ保存している．しかし，目的関数を最小化するx_jは複数存在することが多くあり，その場合最小のx_jのみを選び車の状況を更新し，次の stage で使っている．しかし，選ぶx_jによって車の状況は異なる場合があり，最適なパターンを排除してしまっている可能性がある．つまり，COP で求められる解は局所最適に過ぎない．すべての最適なx_jを考慮することも不可能ではないが，stage を重ねるにつれ計算量が膨大となり現実的でない．しかし，4-3①のように，乗客ベースにすることで PHT の改善が見られており，また，一台当たりの乗車人数が多いほどより改善が顕著となると考えられる．

また，各制限を入れる際，もとの目的関数である乗客旅行時間とのバランスを取るような適当なα，βを使うことが望ましい．これは，過去の需要などから推定される将来需要をもとに設定する必要がある．

5. 結論

本研究では，乗客ベース COP を提案し，ペナルティ関数法を用いて列長の制限と待ち時間の制限を導入した．また，これらの効果をシミュレーションにより検証した．車両ベースと乗客ベースの比較では，乗客ベース COP により PHT が改善していることが示された．各制限については，パラメータα，βを大きくすることでより制限が効き，spillback や個人の極端に長い待ち時間を避けられていることが確認できた．現実世界に適用する際は，予測精度が重要であり，需要予測も組み込むことでより精度を上げられると考えられる．今後は，適切なパラメータα，βの設定方法の検討とともに，ネットワークに拡張し適用可能性を示したい．

参考文献

[1] Sen, S., & Head, K. L. (1997). Controlled optimization of phases at an intersection. Transportation science, 31(1), 5-17

[2] Yu, Z., Xu, G., Gayah, V. V., & Christofa, E. (2020). Incorporating phase rotation into a person-based signal timing optimization algorithm. IEEE Transactions on Intelligent Transportation Systems, 23(1), 513-521.

[3] Wu, Z., Waterson, B., & Anvari, B. (2022). Developing and evaluating a coordinated person-based signal control paradigm in a corridor network. Transportation Planning and Technology, 45(6), 498-523.

[4] Van Katwijk, R. T., De Schutter, B., & Hellendoorn, J. (2007). Look-ahead traffic-adaptive signal control. In Proc. 6th European Congress on Intelligent Transport Systems and Services (ITS'07).

[5] 瀬尾亨. マクロ交通流シミュレーション : 数学的基礎理論と Python による実装. コロナ社, 2023

[6] Seo, T. UXsim: An open source macroscopic and mesoscopic traffic simulator in Python-a technical overview. arXiv preprint arXiv: 2309.17114, 2023