観光ナビにおいて必見スポットをよりよい時間帯に訪問可能にするオンサイトプランニング手法

磯田 祥吾[1,2]　日高 真人[1]　松田 裕貴[1,2]　諏訪 博彦[1,2]　安本 慶一[1,2]

概要：観光経路推薦に関して，既存研究の多くは，次訪問スポットのみの満足度に基づいて推薦しているが，次以降に訪れるスポットを考慮に入れていないため，次スポット以降の観光が制限される．次訪問スポットの満足度と，次訪問スポット以降の期待満足度を考えた場合，この二つにトレードオフ関係が発生する可能性がある．また同一スポットを訪問する際に，訪れる時間帯によって満足度が異なるため，より良い時間帯に訪問することが観光全体の満足度向上につながる．これら 2 点を考慮した上で，ユーザに観光ルートを提示するのが望ましいが，我々が知る限り，そのようなシステムは存在しない．本稿では，上記を考慮するため，まず，次スポットの静的観光地コンテキストと動的観光地コンテキスト，次スポット以降に訪問するスポットから得られる期待満足度の 3 つの要素から成るツアースコアを定式化する．ツアースコアが最大となる観光ルート（次訪問スポットとそれ以降に訪問するスポット群の列）を算出する問題は NP 困難であり，オンサイトで準最適解を求めるため，貪欲法をベースとした 3 つのアルゴリズム：(1) 次スポットのみを考慮した貪欲法，(2) 観光時間全体を考慮した貪欲法，(3) 観光時間全体および探索幅を広げた貪欲法の提案を行う．提案アルゴリズムの有用性を調査するため，京都市東山区にある 20 箇所の訪問スポットを対象に 3 つのアルゴリズムを適用することで，出力解が京都のモデルルートと比較して優位であり，それぞれ 1.9±0.1 (s)，15.9±0.5 (s)，7766.6±8.9 (s) の計算時間で算出できることを確認した．

An on-site planning method in tour navigation for enabling visit to must-see spots at better times

Shogo Isoda[1,2]　Masato Hidaka[1]　Yuki Matsuda[1,2]　Hirohiko Suwa[1,2]　Keiichi Yasumoto[1,2]

1. はじめに

近年，観光産業の需要が高まってきており，個人 1 回あたりの旅行費用も増加傾向にある [1]．それに伴い，個人に最適化された訪問スポット推薦（観光ナビゲーション）の研究が盛んに行われている [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12]．しかし実際に観光してみると，想定外の事象が発生するケースが多い．想定外の事象とは，突発的な豪雨や混雑，イベント，臨時休業などを指す．このような事象により観光に満足できていない観光客もいる．本稿において，このような観光地の状況，環境などの総体を観光地コンテキストと定義する．観光地コンテキストは，訪問スポットの位置，営業時間，価格，特徴など基本的には変化しない静的観光地コンテキストと，天候，混雑度，イベント，臨時休業などの動的観光地コンテキストとの 2 つに大別される [13]．

[1] 奈良先端科学技術大学院大学
Nara Institute of Science and Technology
[2] 理化学研究所 革新知能統合研究センター
RIKEN Center for Advanced Intelligence Project

既存システムの多くは，静的観光地コンテキストに基づいて推薦している．そのため，観光前に調べても，観光中に調べても推薦結果は変化しない．しかしながら，観光地の状況は動的に変化する．例えば，訪問したスポットが臨時休業で観光できなければ満足度は得られない．また，同じ観光スポットであっても，天気の良し悪し，混雑の程度，ライトアップなどのイベントの有無により，満足度は変化する．そのため，各観光スポットの動的観光地コンテキストをリアルタイムに収集または予測し，その結果を考慮に入れた推薦が求められている．

また，既存のシステムの多くは，次に行く観光スポットのみの満足度に基づいて推薦している．しかしながら，仮に満足度の高い観光スポットが離れた場所にある場合，移動に時間がかかるため，そのスポット以降の観光が制限される（訪問できるスポット数が減るなど）ことが考えられる．そのため，観光全体の満足度が低下することが考えられる．満足度が高いスポット一つを訪問するよりも，半分の満足度しか得られないスポットを 3 つ以上訪問する方が観光全体の満足度は高くなることもある．そのため，次に

訪問する観光スポットだけでなく，次以降に訪問可能な観光スポットの期待満足度も考慮した推薦が求められる．

次の観光スポットの満足度と，次以降の観光スポットの期待満足度を考えた場合，この二つにトレードオフ関係が発生する可能性がある．先に示した通り，次の観光スポットとして高い満足度が得られる観光スポットを選択することにより，観光全体の満足度が低くなる可能性がある．また， 同一スポットを訪問する際に，訪れる時間帯によって満足度が異なる可能性がある（夜景が綺麗なスポットなど）．そのスポットにおいて満足度が高い時間帯を推薦することで，観光全体に対する満足度が向上すると考える．そのため，より良い時間帯の訪問スポット推薦が求められる．

そこで本論文では，動的観光地コンテキストと期待満足度を考慮に入れた観光プランニング手法を提案する．観光に対する期待満足度を算出するためには，すべての観光パターンにおける満足度を算出する必要があるが，この問題は NP 困難であり短時間で最適解を導き出すことは現実的ではない．そのため，短時間で準最適解を求めるアルゴリズムを提案する．具体的には，貪欲法をベースとした 3 つのアルゴリズムを設計する．アルゴリズム A（時系列貪欲法）は，次スポットのみを考慮し，最大スコアのスポット上位 3 個を順番に選んでいく貪欲法である．アルゴリズム B（全体単一貪欲法）は，観光時間全体を考慮し，最大スコアのスポット・時間帯の組から順番に上位 3 個を選んでいく貪欲法である．アルゴリズム C（探索幅を考慮した全体貪欲法）は，アルゴリズム B の拡張版であり、選択肢を k 個ずつ木構造で探索していく貪欲法である．

3 つの提案アルゴリズムの有効性を評価するために，京都市東山区にある 20 箇所の観光スポットを対象に 3 つの提案アルゴリズムを適用した．その結果，3 つの提案アルゴリズムは準最適解を，それぞれ 1.9 ± 0.1 (s)，19.2 ± 0.6 (s)，1548.0 ± 55.1 (s) の計算時間で算出できることを確認した．また，アルゴリズム C（探索幅を考慮した全体貪欲法）が最も満足度の高い観光ルートを導出できることが分かった．

2. 関連研究

提案システムでは，収集した観光情報からユーザのプロファイルや観光地の混雑度・天気情報などの観光地コンテキストに応じて推薦すべき訪問スポットを複数提示する．以下では，提案システム実現のために必要な，静的情報を考慮に入れた推薦，動的情報を考慮に入れた推薦，遺伝的アルゴリズムを用いた観光ルート推薦の各手法の既存研究について述べる．

2.1 観光スポットの推薦

一般的な推薦システムは，協調フィルタリングを利用した推薦と内容ベースフィルタリングを利用した推薦の 2 つの手法に分類される [14]．協調フィルタリングとは，多くのユーザの嗜好情報を過去の行動という形で記録し，そのユーザと嗜好の類似度が高い他ユーザの嗜好情報を用いてユーザの嗜好を推測する手法である．また内容ベースフィルタリングとは，アイテムの内容とユーザの嗜好情報を比較し，その関連度に基づいてフィルタリングを行う手法である．近年では，協調フィルタリングと内容ベースフィルタリングを合わせたハイブリッド手法が主流である [15]．ハイブリッド法には，両方の推薦結果を提示するあるいは，何らかの基準に基づいて推薦手法を切り替えるなど多様な手法がある．

2.2 ユーザの嗜好を考慮に入れた推薦

Kwan らは [3] は，PersTour アルゴリズム [4] を用いて，訪問するのに適した POI と各 POI の平均滞在時間を推薦するために，POI の人気度とユーザーの嗜好の両方を考慮した．結果，訪問頻度ベースのユーザの関心と平均的な訪問時間に基づいた従来の手法より，時間ベースのユーザの関心とパーソナライズされた POI カテゴリ訪問時間の方が優位な結果が得られることを明らかにした．

Prarthana ら [5] は，オリエンテーリング問題の実例としてツアー推薦問題をモデル化し，個人に最適化されたツアーを推薦するための 2 つのアルゴリズムを提案した．写真投稿頻度の多いユーザーの嗜好と POI の人気の両方を用いて，適切な POI を推薦する．ユーザ写真投稿頻度データセットから学習されたユーザの嗜好に最適化された旅程を推薦することによって，各ユーザの目標とされた個人最適化レベルを改善している．8 都市の Flickr データセットを使用して，ツアーの人気度，ユーザ関心度などに関してアルゴリズムを評価している．

2.3 動的情報を考慮に入れた推薦

Rohit ら [16] は，さまざまな環境や個人の選択よりも通勤者の好みに応じて最も快適なルートを推奨するための，スマートフォンベースのシステムである ComfRide を提案した．ComfRide の目新しさは，過去のデータおよびコンテキスト情報に基づいて DIOA ベースの構成モデルを利用する路線推薦である．インドの州都で 28 の異なるバス路線で 2 年間の実地試験を行った結果，快適さに対する通勤者の優先順位のさまざまな組み合わせについて，ComfRide の推奨路線は Google ナビゲーションベースの推奨路線よりも平均 30 %優れていた．

Jevinger ら [17] は，交通渋滞の間に起こる新しい状況に適応する個人化された情報を提供することができる旅行プランナーと代替ルートのための提案した．コンテキストをアプリケーションの周囲の状態とし，想定できない事象の発生時に実行する必要がある情報と作業に対処することで，状況に応じた旅行プランナーの分析と開発を支援した．これにより，提供される旅行支援をより個別に最適化するこ

とができ，公共交通機関の混雑のリアルタイムの影響を考慮に入れることで，個人に最適化されたルート提案を行うことができることを明らかにした．

武ら [18] は，天気が確率的に予測できる場合を想定し，任意の天気予報データに対応したスケジュール群を立案する問題を定式化した．欲張り法で出発地点からそれぞれの天候に応じた観光スポットを場当たり的に選び，天候に応じた観光スポットを含むスケジュール群を得て，帰着時刻までに帰着地点に帰ってくるようなスケジュールに変換する．欲張り法で得られたスケジュール群に対して局所探索法 (n-swap 法) を繰り返し適用することによって出力解を改善する．この過程を複数回繰り返し，[19] でのルートと比べて，天気変化を考慮した期待値が高いルートを推薦した．

2.4 遺伝的アルゴリズムを用いた推薦手法

丸山ら [19] は，観光のためのパーソナルナビゲーションシステム “P-Tour” を提案した．P-Tour は，ユーザが出発地と出発時刻，帰着地と帰着時刻，複数の観光候補地と各地への立ち寄り希望度と時間制約（到着時間帯や滞在時間など）を設定すると，制限時間内で巡回可能かつ最も満足度が高くなるような巡回経路（いくつかの観光地を含む）と各観光地への到着・出発予定時刻を含むスケジュールを算出しユーザに提示する機能を提供する機能をもつ．遺伝的アルゴリズムを用いて準最適なスケジュールを高速に算出するアルゴリズムを用いて実装した．

平野ら [20] は，観光によって得られる満足度と，観光客が移動や観光地で消費するリソース（金銭・時間・体力）のトレードオフを考慮に入れた上で，観光客に複数の解（ツアープラン）を提示する方法を提案した．複数の独立した要因間のトレードオフを抱える最適化問題をユーザのリソースの 3 要因と観光満足度の 4 つの要因を独立変数として考慮した多目的最適化問題として定式化した．この問題は NP 困難であるため，現実的な時間で準最適解を求めるため，遺伝的アルゴリズム NSGA-II と 2-opt に基づいたアルゴリズムを実装した．提案アルゴリズムの有効性を評価するために，京都市東山区にある 30 ヵ所の観光スポットを対象にした観光計画に提案アルゴリズムを適用した．その結果，多目的最適解をもつ複数の観光ルート全体を提示するシステムを構築している．

2.5 関連研究の課題および本研究の目的

既存手法では，過去に収集した観光客の移動履歴等のデータを学習することによって，事前に訪問スポットの推薦を行っている．しかし，これらの手法の多くは，混雑度，天気状況や想定外の事象といった観光地コンテキストの動的変化を扱っていない．

また，既存のシステムの多くは，次に行く観光スポットのみの満足度に基づいて推薦している．しかし，これらの手法では，観光全体で最適な推薦になるとは限らない．そのため，次に訪問する観光スポットだけでなく，次以降に訪問可能な観光スポットの期待満足度も考慮した推薦が求められる．

動的観光地コンテキストを考慮した研究 [16], [17], [18] や，観光ルート全体の最適化を目指す研究 [20] も存在するが，オンサイトで，動的観光地コンテキストと期待満足度の両方を考慮した研究は存在しない．

オンサイトで動的観光地コンテキストと期待満足度を考慮に入れた観光プランを提案するためには，次訪問スポットとそれ以降の訪問スポット群からなる全ての観光経路に対する満足度を算出する必要がある．しかしこの問題は NP 困難であり，短時間で最適解を導き出すことは現実的ではない．そのため，本研究では，短時間で準最適解を求めるアルゴリズムを提案することを目的とする．

3. ツアースコアの定式化

本研究では，オンサイトで，次スポットおよびそれ以降のスポット群を訪問する際のツアー全体の評価値を求めるための変数として，以下の，次スポットの静的スコアおよび動的スコア，次スポット以降の事後期待スコアを定義する．訪問可能スポットの集合 S，次訪問スポット s，スポット s への到着時刻を t とした時のツアースコア $Tour(S,s,t)$ を，下記の式で示す．

$$\begin{aligned} Tour(S,s,t) \;=\; & SV(s)+DV(s,t) \\ & +EV(s,S-\{s\},t+time(s)) \end{aligned} \tag{1}$$

上記において，$SV(s)$，$DV(s)$ は，それぞれ，スポット s の静的スコア，動的スコアであり，以下で詳細に定義する．また，$EV(s,S',t')$ は，スポット s を観光後の時刻 t' からスポットの集合 S' のいくつかを観光する際に得られるツアースコアの最大値であり，後に定義する．なお，$time(s)$ は s の観光時間とする．

上記の式から算出されるツアースコアは，観光客がどのスポットを次の訪問スポットとして選ぶかで異なってくる．次訪問スポットとして，複数の観光スポットを考慮し，それぞれツアースコアを算出・提示し，観光客に次訪問スポット選択させることが本研究の主題である．

3.1 観光スポットの静的スコア

本研究において，ユーザの嗜好と訪問スポット s のマッチング度合いを静的スコア $SV(s)$ と定義する．静的スコアは，Kwan Hui ら [3] の評価手法を用いる．

3.2 観光スポットの動的スコア

スポットの動的スコア $DV(s)$ は，下記の式で算出する．

$$DV(s) = TV(s,t)+CE(s,t)+WE(s,t) \tag{2}$$

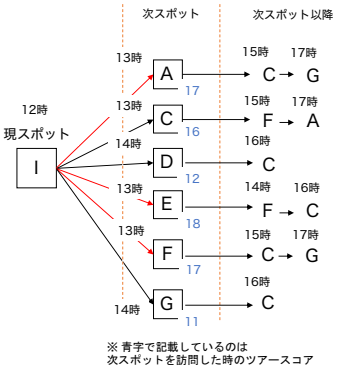

図 1: 時系列貪欲法
（アルゴリズム A）

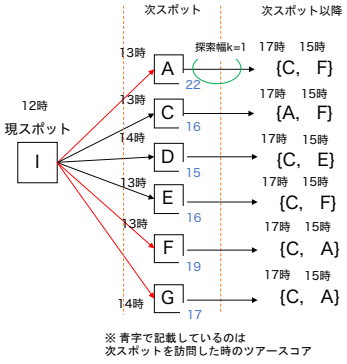

図 2: 全体単一貪欲法
（アルゴリズム B）

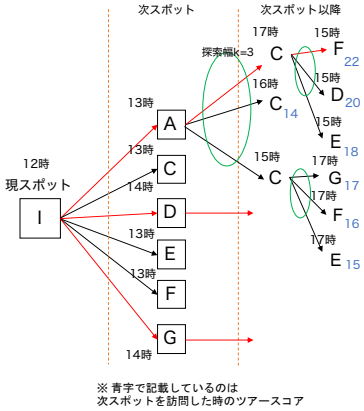

図 3: 探索幅を考慮した全体貪欲法
（アルゴリズム C）

上記において，$TV(s,t)$ は，スポット s の時刻 t における加算ポイントであり，例えば夕陽が美しいスポットや夜景がメインのスポットでは，それぞれ t が夕刻，夜であるときに正の値をとる．$CE(s,t)$ は，混雑度による加算ポイントであり，時刻 t においてスポット s が混雑しているときに負の値をとる．$WE(s,t)$ は時刻 t におけるスポット s での天候による加算ポイントであり，スポットのタイプと天候により正から負の値をとる．これらの値は，ユーザの嗜好，スポット，状況ごとに用意されているものとする．

3.3 事後期待スコア

スポット s を訪問後の事後期待スコア $EV(s,S',t')$ は，下記の式で再帰的に定義する．

$$EV(s,S',t') = \begin{cases} 0 & (\text{if } t' \geq T_{end}) \\ EV_{\max} & (otherwise) \end{cases} \quad (3)$$

$$EV_{\max} = \max_{s' \in S' \wedge movet(s,s') \leq T_{move}} \Big(SV(s') + DV(s',t') + EV\big(S' - \{s'\}, t' + movet(s,s') + stayt(s')\big) \Big) \quad (4)$$

上記において，T_{end} は，観光終了時刻，T_{move} はスポット間の移動時間の上限，$movet(s,s')$ はスポット s から s' への移動時間，$stayt(s')$ はスポット s' における滞在時間とする．

事後期待スコア $EV(s,S',t')$ は，スポット s を訪れたあと，観光終了時刻 T_{end} までに訪問可能なスポット群で得られるスコアの最高値として定義している．ただし，スポット s から移動可能なスポット s' として，移動時間が T_{move} 以内という制約を置いている．

4. オンサイト観光経路探索アルゴリズム

本章では，本研究の主題であるツアースコアを算出するにあたって，3 つのアルゴリズム，及びその詳細について示す．各アルゴリズムで用いる想定環境の説明を表 1 に示す．表 1 の訪問予定スポットリスト Z とは，アルゴリズム

表 1: アルゴリズムで用いる想定環境

定義	説明
全スポット集合 S_{all}	$A, B, C, D, E, F, G, H, I$
訪問済みスポット集合 $S_{visited}$	B, H
未訪問スポット集合 S	A, C, D, E, F, G
観光時間 T	13 時～18 時
タイムスロット幅 tl	1 時間
現在地 cp	I
現在時刻 ct	12 時
訪問予定スポットリスト Z	$[ct, cp, 0]$

表 2: アルゴリズムで用いる想定環境のスコア値

		Time					
		13:00	14:00	15:00	16:00	17:00	18:00
Spot	A	7	3	4	5	6	7
	B	4	5	3	2	4	5
	C	4	5	6	7	9	6
	D	4	5	4	3	2	6
	E	4	3	2	1	2	3
	F	7	7	6	4	3	2
	G	5	4	3	2	4	7
	H	4	5	4	3	2	1
	I	2	1	4	5	1	6

を適用したときに訪問することができるスポットを格納するためのリストである．このリスト内には，各スポットごとに { 到着時刻, スポット ID, 得られる満足度 } が格納されている．訪問予定スポットリスト Z の合計満足度が，本研究主題のツアースコアである．本稿では，ツアースコアの上位 3 位をユーザに提示するために算出する．

4.1 NP 困難を回避するための時間幅の設定

実観光を想定した時，各スポットの全時刻（例えば 1 分単位）に対して評価値を持つことで，高い精度で観光ルー

ト推薦を行える可能性がある．しかし，観光ルートの推薦は観光スポットの組み合わせの計算量が膨大であり，NP困難である．加えて本アルゴリズムでは，今後期待できる満足度を算出する必要があるため，通常の観光スケジューリング問題よりもさらに計算時間を要すると考えられる．そこで，各スポットに対して 1 分ごとに評価値を持つのではなく，一定時間幅（想定環境では 1 時間）に 1 つだけのスポットを訪問するとの仮定の下，一定時間幅ごとに評価値を持つことで．短時間で準最適解を算出することができるようにする．そのため提案手法では，一定時間幅の時間帯ごとにスポットの評価値を，静的コンテキストと動的コンテキストの和として算出する．静的スコア $SV(s)$ は時間帯による変化はないが，動的コンテキスト $DV(s,t)$ は時刻 t に伴う評価値の変動があるため，各時間帯の各スポットにおける評価値は異なる．各アルゴリズムの説明を行うために，想定環境でのスコア値を表 2 に示す．表 2 を用いて，各アルゴリズムの実行例を説明する．

4.2 アルゴリズムの概要

本研究の主題であるツアースコアを算出するために，3 つのアルゴリズム（時系列貪欲法，全体単一貪欲法，探索幅を考慮した全体貪欲法）を提案する．

時系列貪欲法（アルゴリズム A） 次スポットで得られる評価値のみを考慮した貪欲法である．想定環境に適用した時の概略を図 1 に示す．この手法では，現在地から未訪問スポット集合 S の各スポットまでの到着時刻を，滞在時間および移動時間を考慮して算出する．到着時刻の評価値が最大のスポットを選択していき，ツアースコアが上位 3 位まで（$k=3$ を想定）の観光ルートを決定する手法である．

全体単一貪欲法（アルゴリズム B） 全ての時間帯で得られる評価値を考慮した貪欲法である．想定環境に適用した時の概略を図 2 に示す．この手法では，訪問予定スポットリスト Z の各スポットとの移動時間および滞在時間を考慮して，観光全体時間の中で評価値が最大のスポット・時間帯の組を選択していき，ツアースコアが上位 3 位までの観光ルートを決定する手法である．

探索幅を考慮した全体貪欲法（アルゴリズム C） 全ての時間帯の中で上位 k 位以内までの得られる評価値を考慮した貪欲法である．想定環境に適用した時の概略を図 3 に示す．この手法では，訪問予定スポットリスト Z の各スポットとの移動時間および滞在時間を考慮して，再帰的に観光全体時間の中で評価値が上位 k 位以内のスポットを選択していき，ツアースコアが上位 3 位の観光ルートを決定する手法である．

次節以降では，各アルゴリズムの詳細を擬似コードを用いて述べるとともに，アルゴリズムを想定環境に適用した時の実行例について説明する．

Algorithm 1: アルゴリズム A（時系列貪欲法）

```
P1: Main()
    input          : S_all, S_visited, T, sp, cp, ct
1   S = S_all \ S_visited
2   k = 3
3   Z = {cp}
4   Recommend_Route = GetOptRoutes
      (k, cp, ct, T, S, Z)
5   ShowRecommendation (Recommend_Route)

P2: GetOptRoutes()
    input          : k, cp, ct, T, S, Z
    output         : Recommend_Route
1   TourScore_route ← { }
2   i = 0
3   while S ≠ φ do
4       Z.seq ← Insert(Z.seq, S[i].spot)
5       Z.score ← Z.score + S[i].score
6       S_remain ← S\S[i].spot
7       R_out = GetEVRoutes(T, S_remain, Z)
8       TourScore_route ←
          AddRoute(TourScore_Route, R_out)
9       i ← i + 1
10  Recommend_Route =
      GetMaxTourScoreRoute(k, TourScore_route)
    return Recommend_Route

P3: GetEVRoutes()
    input          : T, S_remain, Z
    output         : R_out
    temporal_variable: Z_tmp, Z_out
1   Z_tmp ← Z
2   ps.staytime ←
      GetTimeSlotForStaying(Z_tmp[−1])
3   list_spots ← {}
4   j = 0
5   while j < Size(S_remain) do
6       ns ← S_remain[j]
7       ns.movingtime ←
          GetTimeSlotForMoving(ps, ns, ps.staytime)
8       ns.staytime ← GetTimeSlotForStaying(ns)
9       j ← j + 1
10      if ns.movingtime + ns.staytime ≤ T.end then
11          list_spots ← AddSpot(list_spots, ns)
12  if list_spots ≠ φ then
13      Z_tmp.seq ← AddSpot(max(list_spots))
14      R_out ← GetEVRoutes(T, S_remain, Z_tmp)
15      if R_out ≠ φ then
16          return R_out
17      else
18          break
19  return R_out
```

4.3 アルゴリズム A（時系列貪欲法）

4.3.1 アルゴリズム A の詳細

本節では、アルゴリズム A について提案する．擬似コードを Algorithm 1 に示す．

Algorithm 1 - P1（Main）では，Algorithm 1 - P2（GetOptRoutes）で算出した次スポットの評価値と Algorithm 1 - P3（GetEVRoutes）で算出した事後期待スコアの和であるツアースコアが上位 3 位のルート（訪問予定スポットリスト）の結果および各ルートのツアースコアを出力する．

Algorithm 1 - P2 では，未訪問スポット集合 S の中の各スポットの評価値（静的スコアと動的スコアの和）を算出する．この時，選択したスポットを訪問予定スポットリスト Z に格納する．次に更新した未訪問スポット集合 S_{remain}，訪問予定スポットリスト Z，観光時間 T を引数として，Algorithm 1 - P3 において，ツアースコアを算出する．ここで，各次スポットのツアースコアを算出した中で，上位 3 つのルートを提示する関数を GetMaxTourScoreRoute とする．

Algorithm 1 - P3 では，訪問予定スポットリスト Z に格納されたスポットの中で，最も観光時刻が遅いスポットを選択する．選択したスポットから，未訪問スポット集合 S_{remain} の各スポットに対して，移動時間及び各スポットの滞在時間を考慮した，到着時刻における評価値を算出する．各スポットそれぞれの評価値を算出して，算出した評価値の中で最も高い評価値のスポットを，訪問予定スポットリスト Z に追加する．訪問予定スポットリスト Z に追加したスポットは未訪問スポット集合 S_{remain} から削除する．同様の処理を観光終了時刻 $T.end$ まで，あるいは未訪問スポット集合が空になる時点まで繰り返し行う．終了時点での訪問予定スポットリスト Z に格納されているスポットの合計評価値が，ツアースコアとなる．ここで各スポットの移動時間，および滞在時間を算出する関数を GetTimeSlotForMoving，GetTimeSlotForStaying とする．

4.3.2 アルゴリズム A の実行例

アルゴリズム A を想定環境に適用した時の実行例を示す（表 2 の設定に沿って説明する）．開始時はユーザの現在地は {12 時, I} である．未訪問スポット集合 S_{remain} の各スポットへの移動時間を考慮した評価値を算出する．例えば，次スポットを {13 時, A} に訪問する時，得られる満足度は 7 である．この時の {13 時, A, 7} を訪問予定スポットリスト Z に格納する．次に，{13 時, A} での滞在時間および各スポットとの移動時間を考慮すると，{15 時, C, 6} が最大値であるため，訪問予定スポットリスト Z に格納する．{15 時, C} での滞在時間および各スポットとの移動時間を考慮すると，{17 時, G, 4} が最大値であるため，訪問予定スポットリスト Z に格納する．スポット G を格納し，{17 時, G} での滞在時間により観光終了となる．この時の終了時点での訪問予定スポットリスト Z に格納されている満足度の合計は 17 であり，この値がツアースコアである．各未訪問スポット集合 S_{remain} の各スポットのツアースコアを算出し，ツアースコア上位 3 位をユーザに提示する．

4.4 アルゴリズム B（全体単一貪欲法）・C（探索幅を考慮した全体貪欲法）

4.4.1 アルゴリズム B・C の詳細

本節では、アルゴリズム B・C について提案する．なお、アルゴリズム B・C は探索幅 k が異なる（それぞれ、探索幅 $k=1$，探索幅 $k>1$）ということ以外は共通であることから、同一の擬似コード Algorithm 2 を用いて説明する．

Algorithm 2 - P1（Main）は，Algorithm 1 - P1 と同一であるため、4.3.1 項を参照されたい．

Algorithm 2 - P2（GetOptRoutes）では，未訪問スポット集合 S の中の各スポットの評価値（静的スコアと動的スコアの和）を算出する．この時，選択したスポットを訪問予定スポットリスト Z に格納する．次に探索幅 k，現在地 cp，未訪問スポット集合 S，訪問予定スポットリスト Z，観光時間 T を引数として，後述の Algorithm 2 - P3（GetEVRoutes）において，ツアースコアを算出する．

Algorithm 2 - P3 では，Algorithm 2 - P4（SortByScore）から算出したソート結果 TS で評価値が上位 k 個の { 到着時刻，スポット } を選択し，選択した { 到着時刻，スポット } を Algorithm 2 - P5（IsReachable）で移動時間と滞在時間を考慮に入れて，訪問可能と判定した場合，{ 到着時刻，スポット } を仮訪問予定スポットリスト Z_{tmp} に格納する．次に探索幅 k，現在地 cp，未訪問スポット集合 S_{remain}，複製された仮訪問予定スポットリスト Z_{tmp}，観光時間 T を引数として，再帰的に GetEVRoutes を行う．再帰結果が複数帰ってきた場合，その中でベストな評価値を持つルートを Z_{out} に格納する．もし再帰結果がなかった場合，再帰前の結果を Z_{out} に格納する．このときに格納した Z_{out} を R_{out} に格納し，R_{out} に k 個のルートが格納されていた場合は繰り返し処理を終了し，その時の R_{out} を返す．終了時点での一時的訪問予定スポットリスト R_{out} に格納されているスポットの合計評価値が，ツアースコアとなる．また，複数の再帰の返り値でベストな評価値を持つルートを返す関数を GetMaxRecrusive とする．

Algorithm 2 - P4 では，観光時間 T，未訪問スポット集合 S，訪問予定スポットリスト Z をもとに降順ソートを行う．ソート結果を Algorithm 2 - P3 に返す．また，各スポットの移動時間，および滞在時間を算出する関数を GetTimeSlotForMoving，GetTimeSlotForStaying とする．

Algorithm 2 - P5 では，選択した { 到着時刻，スポット } が，訪問予定スポットリスト Z と比較して訪問できるかどうかを，移動時間および滞在時間を考慮して判定する．

上記の擬似コードで取り扱った探索幅 k について，アル

ゴリズム B では $k=1$，アルゴリズム C では $k>1$ である．本稿では，アルゴリズム C の探索幅を $k=3$ と設定した．

4.4.2 アルゴリズム B・C の実行例

アルゴリズム B を想定環境に適用した時の実行例を下記に示す（表 2 の設定に沿って説明する）．ユーザの現在地が $\{12\text{ 時}, I\}$ であり，未訪問スポット集合 S の各スポットへの移動時間を考慮した評価値を算出する．例えば，次スポットを $\{13\text{ 時}, A\}$ に訪問する時，得られる満足度は 7 である．$\{13\text{ 時}, A, 7\}$ を訪問予定スポットリスト Z に格納する．次に，スポット A および，スポット A での滞在時間を除いた観光全体時間の中で最も大きい評価値は 9 である．その時の到着時刻のスポットは $\{17\text{ 時}, C\}$ である．$\{13\text{ 時}, A\}$ から $\{17\text{ 時}, C\}$ は移動時間および滞在時間を考慮すると，訪問可能であるため，訪問予定スポットリスト Z に格納する．次に，観光全体時間の中で最も大きい評価値は 7 である．しかし，訪問予定スポットリスト Z に格納している { 到着時刻, スポット } と比較すると，評価値 7 を持つどのスポットも訪問することはできない．次に大きい評価値は 6 である．その時の到着時刻のスポットは $\{15\text{ 時}, F\}$ である．移動時間および滞在時間を考慮すると，$\{13\text{ 時}, A\}$ から $\{15\text{ 時}, F\}$ まは訪問可能であり，$\{15\text{ 時}, F\}$ から $\{17\text{ 時}, C\}$ は訪問可能であるため，訪問予定スポットリスト Z に格納する．訪問予定スポットリスト Z に，これ以上選択することができないため観光終了となる．この時の終了時点での訪問予定スポットリスト Z に格納されている満足度の合計は 22 であり，この値がツアースコアである．未訪問スポット集合 S の各スポットのツアースコアを算出し，ツアースコア上位 3 位をユーザに提示する．

アルゴリズム C を想定環境に適用した時の実行例を下記に示す．ユーザの現在地が $\{12\text{ 時}, I\}$ であり，未訪問スポット集合 S の各スポットへの移動時間を考慮した評価値を算出する．例えば，$\{13\text{ 時}, A\}$ に訪問する時，得られる満足度は 7 である．$\{13\text{ 時}, A, 7\}$ を訪問予定スポットリスト Z に格納する．次にスポット A および，スポット A での滞在時間を除いた観光全体時間の中で上位 3 位の評価値は，$\{17\text{ 時}, C, 9\}$，$\{16\text{ 時}, C, 7\}$，$\{15\text{ 時}, C, 6\}$ である．まず，$\{17\text{ 時}, C, 9\}$ は，移動時間および滞在時間を考慮すると，$\{13\text{ 時}, A\}$ から $\{17\text{ 時}, C\}$ は訪問可能であるため，仮訪問予定スポットリスト Z_{tmp} に格納する．次にスポット A からスポット C までの移動時間およびスポット C での滞在時間を考慮に入れた上での上位 3 位の評価値は，$\{15\text{ 時}, F, 6\}$，$\{15\text{ 時}, D, 4\}$，$\{15\text{ 時}, E, 2\}$ である．$\{13\text{ 時}, A\}$ から訪問可能であり，$\{17\text{ 時}, C\}$ への訪問も可能である．訪問予定スポットリストに格納した段階で，どのルートも観光終了時間となる．各評価値をもとにツアースコアを算出した場合，このルートでは $\{15\text{ 時}, F, 6\}$ を仮訪問予定スポットリスト Z_{tmp} に加えると最大ルートを訪問することができるため，この最大ルートを R_{out} に追加する．同様の

Algorithm 2: アルゴリズム B（全体単一貪欲法）・C（探索幅を考慮した全体貪欲法）

P1: `Main()`
input : $S_{all}, S_{visited}, T, sp, cp, ct$

1 $S = S_{all} \setminus S_{visited}$
2 $k = 3$
3 $Z = \{cp\}$
4 $Recommend\_Route$ = `GetOptRoutes` (k, cp, ct, T, S, Z)
5 `ShowRecommendation` $(Recommend\_Route)$

P2: `GetOptRoutes()`
input : k, cp, T, S, Z
output : $Recommend\_Route$

1 $TourScore\_route \leftarrow \{\}$
2 $i = 0$
3 **while** $S \neq \phi$ **do**
4 $arrival\_time = ct +$ `PathTime` $(cp, S[i])$
5 $Z.seq \leftarrow$ `Insert` $(Z.seq, S[i])$
6 $Z.score \leftarrow Z.score + S[i][arrival\_time].score$
7 $R_{out} =$ `GetEVRoutes` (k, cp, T, S, Z)
8 **if** $R_{out} \neq \phi$ **then**
9 $maxEVRoute =$ `GetMaxEVRoute` (R_{out})
10 $TourScore\_route =$ `AddRoute` $(TourScore\_route, maxEVRoute)$
11 **else**
12 $TourScore\_route =$ `AddRoute` $(TourScore\_route, Z)$
13 $i \leftarrow i + 1$
14 $Recommend\_Route =$ `GetMaxTourScoreRoute` $(TourScore\_route)$
15 `return` $Recommend\_Route$

P3: `GetEVRoutes()`
input : k, cp, T, S, Z
output : R_{out}
temporal_variable: $Z_{tmp}, Z_{out}, R_{result}$

1 $TS \leftarrow$ `SortByScore` (T, S, Z)
2 $R_{out} \leftarrow \{\}$
3 $j = 0$
4 **while** $i <$ `Size` (TS) **do**
5 **if** `IsReachable` $(Z, TS[j])$ **then**
6 $Z_{tmp}.seq \leftarrow$ `Insert` $(Z.seq, TS[j].spot)$
7 $Z_{tmp}.score \leftarrow Z_{tmp}.score + TS[j].score$
8 $R_{result} \leftarrow$ `GetEVRoutes` (k, cp, T, S, Z_{tmp})
9 **if** $R_{result} \neq \phi$ **then**
10 $Z_{out} \leftarrow$ `GetMaxRecrusive` (R_{result})
11 **else**
12 $Z_{out} \leftarrow Z_{tmp}$
13 $R_{out} \leftarrow$ `AddRoute` (R_{out}, Z_{out})
14 **if** `Size` $(R_{out}) == k$ **then**
15 `break`
16 $j \leftarrow j + 1$
17 `return` R_{out}

アルゴリズム B・C（続き）

P4: `SortByScore()`
input : T, S, Z
output: TS

$T_{remain} = T \backslash (\texttt{GetTimeSlotForStaying}(Z) \cup \texttt{GetTimeSlotForMoving}(cp, Z))$
$S_{remain} = S \backslash Z$
$TS = T_{remain} \times S_{remain}$
`return` TS

P5: `IsReachable()`
input : TS, Z
output: $move\_bool$

$move\_bool = False$
while js *in* Z **do**
  if $abs(\texttt{PathTime}(TS[i].spot, js)) \leq abs(TS[i].time - js.time)$ **then**
    $move\_bool = True$
  else
    $move\_bool = False$
    `return` $move\_bool$
`return` $move\_bool$

ことを {16 時, C, 7}，{15 時, C, 6} でも行う．R_{out} のルートが 3 つになった時点での，R_{out} の中での最大値ルートを $Tour\_Score\_Route$ に追加する．この時点で {13 時，A} に訪問した時のツアースコアを算出することができる．各未訪問スポット集合 S の各スポットに上記と同様の処理を行い，それぞれのツアースコアを算出し，ツアースコア上位 3 位をユーザに提示する．

5. 実験概要

5.1 実験目的

本稿では，提案アルゴリズムを実際の京都府京都市東山区にある 20 箇所の PoI を含むエリア（図 4）に対して適用することで，アルゴリズムの出力する解（ツアースコア）についての評価を行う．また，本研究ではオンサイトナビゲーションを想定しているため，ツアースコアを出力するまでの計算時間が現実的であるかどうかの評価を行う．

5.2 実験内容

本実験では，実際にアルゴリズムを京都府京都市東山区にある 20 箇所の PoI を含むエリアに対して適用する．アルゴリズムは Python 言語にて記述し，CPU：Intel Core i5 2.3GHz，メモリ：8.0GB，OS：macOS Catalina のスペックを持つマシンで実行する．

各スポット間の移動時間，各スポットの滞在時間については，観光情報誌 [21], [22] 及び，観光情報誌に記載されていない値に関しては Google Map API から取得したデータを用いる．静的スコア，動的スコアは Google Map から

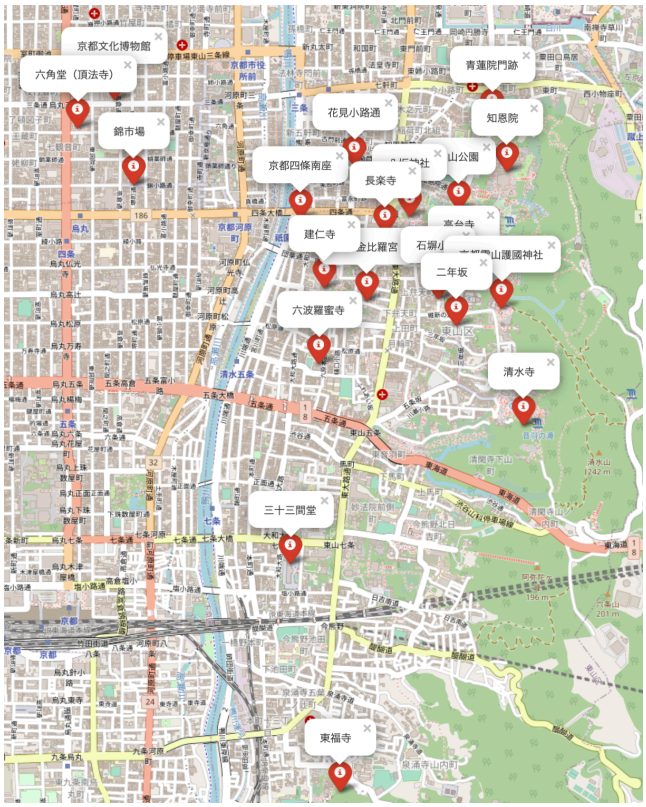

図 4: 東山 (祇園) エリアの PoI のプロット

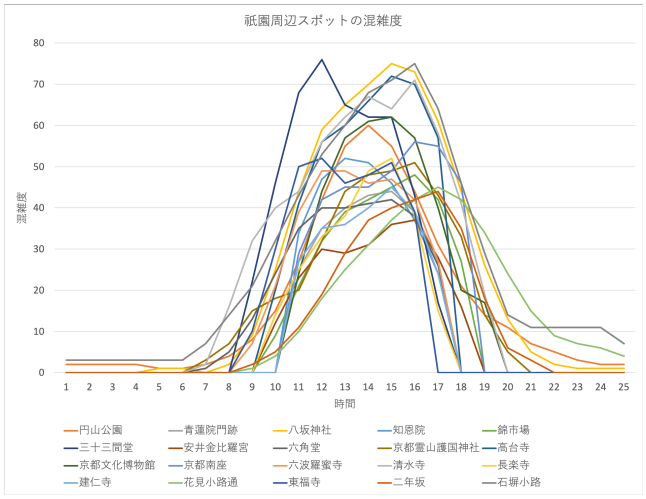

図 5: 東山 (祇園) エリアの PoI の混雑度

取得したデータを用いる．静的スコア SV は各スポットに対して取得できる人気度を 1 から 5 にスケール変換し用いる．動的スコアの混雑度 CE に関して，2020 年 4 月 2 日時点で検索した日曜日における各スポットの混雑度を取得する．各スポットの混雑度を可視化した図を図 5 に示す．取得した混雑度は時間幅 30 分毎の値である．10 分毎の値を用意するため，30 分毎の値を複製する．複製した値の混雑度の逆数をとり，0 から 2 のスケール変換を行い用いる．加算ポイント TV に関しては，0 から 2 のスケールとする．3 つのスポット（高台寺，清水寺，知恩院）に対しては，17 : 30 – 21 : 00 は +2 を付与する．これは観光情報誌で写

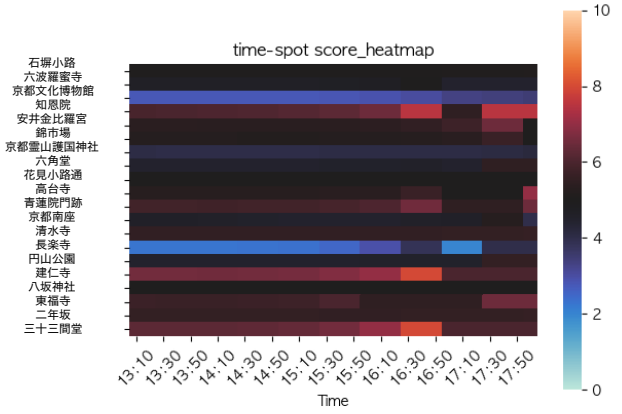

図 6: 各時間帯の各スポットにおけるスコア

真付きで紅葉のライトアップが大きく特集されているため +2 を付与する．4 つのスポット（六角堂，円山公園，青蓮門跡，東福寺）に対しては，観光情報誌で写真なしで紅葉のライトアップが記載されていたため +1 を付与する．他のスポットにおいては，0 とする．天候情報 WE に関しては，−1 から +1 のスケールとする．屋外のスポットの場合，雨の場合，−1 を付与し，晴れなら 1 とする．屋内のスポットは雨の場合，+1 を付与し，晴れなら 0 とする．各スポットの各時間帯に対して，静的スコアと動的スコアから算出した合計スコアのヒートマップを図 6 に示す．

本提案アルゴリズムを適用することで，上記の静的スコア，動的スコア，事後期待スコアからツアースコアを算出して評価する際，観光情報誌に記載されているモデルルートの合計値と比較することで，出力解が優れているかどうかの検証を行う．また計算時間はオンサイトナビを想定しているため，3 分以内の出力が望ましいと考え評価を行う．

6. 実験結果

実験環境として，季節を秋とし，天候は晴れを想定する．また観光時間は 13 時から 18 時までの 5 時間を想定したものとする．出発地点は祇園四条駅とし，訪問済みスポットは無しとする．実験環境での 3 つのアルゴリズムを 5 回実行し，その時の出力解および計算時間を下記に示す．

6.1 出力解

表 3 は，各アルゴリズムを実験環境に適用した時の，上位 3 位の観光ルートの出力解を示す．ツアースコアの比較を行うと，アルゴリズム C が最も優れており，アルゴリズム B が最も劣っていた．まず，アルゴリズム A とアルゴリズム B を比べたときに，アルゴリズム A では，観光全体時間の中で評価値が最大のスポット（16 時半，三十三間堂）あるいは（16 時半，建仁寺）を訪問することができない．アルゴリズム A では，行き当たりばったりに評価値が大きいスポットを訪問するために，滞在時間及び移動時間を考慮に入れると，人気スポットを最も評価値が高くなる時間帯に訪問できなくなる現象が生じる．そのため，アルゴリズム B の方が，各スポットをより良い時間帯に訪問できる可能性が高いと言える．しかし期待満足度の観点では，アルゴリズム A の方が優位な結果であった．アルゴリズム A とアルゴリズム C を比べたときに，訪れるスポットにあまり変化はないが，訪れる時間帯が異なっている．全体の評価値の中で大きいスポット上位 k 番目までを考慮しているアルゴリズム C の方が，より良い時間帯にスポットを訪れることができ，かつ期待満足度が優位な結果であった．アルゴリズム B とアルゴリズム C を比べたときに，アルゴリズム C は全体の評価値の中で大きいスポット上位 k 番目までを考慮し，かつ今後訪問できるスポットをも考慮することができるため，ツアースコアが優位な結果であった．しかしアルゴリズム B では，建仁寺を訪問する時のスコア値が優位な結果となった．

以上より，3 つのアルゴリズムは，各アルゴリズムの目的に対して有効に機能していることが言える．一方で，アルゴリズム B では，他のアルゴリズムに比べて，スコアが最も高い時間帯のスポットを訪れることができるため，単純に優劣を比較することができない．

表 4: 各アルゴリズムの計算時間

	計算時間 (s)					
	1 回目	2 回目	3 回目	4 回目	5 回目	Mean
アルゴリズム A	1.8	2.0	1.9	1.8	1.8	1.9 ± 0.1
アルゴリズム B	16.5	15.5	15.4	16.6	15.7	15.9 ± 0.5
アルゴリズム C	7770.3	7751.6	7762.2	7772.5	7776.6	7766.6 ± 8.9

6.2 計算時間

各アルゴリズムを 1 回目から 5 回目の計算時間を表 4 に示す．この結果から，各アルゴリズムはそれぞれ 1.9 ± 0.1 (s)，15.9 ± 0.5 (s)，7766.6 ± 8.9 (s) の平均計算時間で結果を出力できることがわかった．アルゴリズム A，B は計算時間として実用的であり，アルゴリズムの性能として満足のいくものである．一方，アルゴリズム C は計算時間として，非現実的であるが，出力解（ツアースコア）の性能が最も良いものとなった．なお本手法では，観光客がオンサイトでの使用を想定としているため，アルゴリズム A と B は，オンサイトでの使用を可能としているが，アルゴリズム C はオンサイトでの使用には，現段階では可能でないため，計算時間を大幅に（例えば 3 分以内程度）短縮する必要があり，それは今後の課題である．

6.3 アルゴリズム C における幅の設定

本稿では，アルゴリズム C において探索幅 $k = 3$ と設定している．本節では，探索幅 k を可変的に設定し，探索幅 k の妥当性を検討する．図 7 は，探索幅 $k = 1$ から 5 まで

表 3: 各アルゴリズムの結果

	Route	Result	Tour_Score	Count_Spot	Mean
アルゴリズム A	最大値	[13:20, 高台寺, 5.3], [14:00, 建仁寺, 6.5], [14:50, 三十三間堂, 6.3], [16:10, 知恩院, 6.5], [16:50, 青蓮院門跡, 6.5], [17:30, 円山公園, 5.6], [17:50, 石塀小路, 5.1]	41.9	7	41.6
	二番目	[13:10, 建仁寺, 6.6], [14:00, 三十三間堂, 6.3], [15:20, 知恩院, 6.1], [16:00, 青蓮院門跡, 6.0], [16:50, 高台寺, 5.7], [17:30, 円山公園, 5.6], [17:50, 石塀小路, 5.1]	41.4	7	
	三番目	[13:20, 三十三間堂, 6.3], [14:30, 建仁寺, 6.6], [15:20, 知恩院, 6.1], [16:00, 青蓮院門跡, 6.0], [16:50, 高台寺, 5.7], [17:30, 円山公園, 5.6], [17:50, 石塀小路, 5.1]	41.4	7	
アルゴリズム B	最大値	[13:10, 花見小路通, 5.0] , [13:40, 青蓮院門跡, 5.8] ,[14:30, 石塀小路, 5.1], [15:00, 三十三間堂, 6.4], [16:30, 建仁寺, 8.0], [17:30, 知恩院, 7.5]	37.8	6	37.8
	二番目	[13:10, 石塀小路, 5.1], [13:40, 青蓮院門跡, 5.8], [14:30, 石塀小路, 5.1], [15:00, 三十三間堂, 6.4], [16:30, 建仁寺, 8.0], [17:30, 知恩院, 7.5]	37.8	6	
	三番目	[13:10, 八坂神社, 5.0] , [13:40, 青蓮院門跡, 5.8], [14:30, 石塀小路, 5.1], [15:00, 三十三間堂, 6.4], [16:30, 建仁寺, 8.0], [17:30, 知恩院, 7.5]	37.8	6	
アルゴリズム C	最大値	[13:10, 八坂神社, 5.0], [13:40, 石塀小路, 5.1], [14:00, 花見小路通, 5.0], [14:20, 円山公園, 4.5], [14:40, 青蓮院門跡, 5.8], [15:30, 三十三間堂, 6.6], [16:40, 建仁寺, 8.0], [17:30, 知恩院, 7.5]	47.5	8	47.4
	二番目	[13:10, 花見小路通, 5.0], [13:30, 八坂神社, 5.0], [14:00, 青蓮院門跡, 5.8], [14:40, 知恩院, 6.1], [15:20, 石塀小路, 5.1], [15:40, 建仁寺, 6.8], [16:30, 三十三間堂, 8.0], [17:50, 円山公園, 5.6]	47.3	8	
	三番目	[13:10, 石塀小路, 5.1] , [13:30, 八坂神社, 5.0], [14:00, 青蓮院門跡, 5.8], [14:40, 知恩院, 6.1], [15:20, 石塀小路, 5.1], [15:40, 建仁寺, 6.8], [16:30, 三十三間堂, 8.0], [17:50, 円山公園, 5.6]	47.3	8	

変更した時の，計算時間及びツアースコア（観光全体の満足度）を示す．図 7 の結果より，計算時間は指数関数的に増加していき，ツアースコアは段階的に大きくなっていることがわかる．これは探索幅の増加に伴い，今後訪問できるスポット・時間帯のより多くの組み合わせを探索できるためである．

6.4 モデルルートとの比較

本節では，出力解が実際の観光情報誌 [21], [22] で取り扱われているルートと比較することで，アルゴリズムの評価を行う．表 5 は，京都の観光情報誌に記載されている想定時間 5 時間の東山（祇園）エリアの観光スポットのモデルルートを抜粋し，実験環境においての評価値をもとに算出したものである．モデルルート 1（スコア：27.3）と表 3 の各アルゴリズムのツアースコア（スコア：41.6，37.8，47.4）を比較した時，本アルゴリズムのツアースコアの結果が優れていることがわかる．この要因として，モデルルート 1

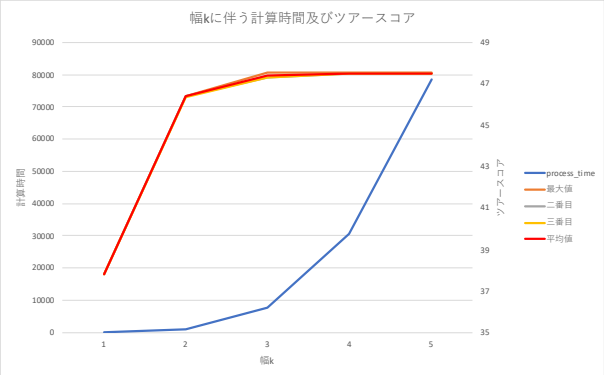

図 7: 探索幅 k に伴う計算時間およびツアースコア

では，観光情報誌で取り上げられている有名スポット（滞在時間が長いスポット）を多く選択しているかつ評価値が高い時間帯（より良い時間帯）に訪れていないことから，ツアースコアが小さくなっている．モデルルート 2（スコア：39.3）と各アルゴリズムのツアースコア（スコア：

表 5: モデルルートのスコア

Model_route	Result	Tour_Score	Count_Spot
モデルコース 1	[13:20, 清水寺, 5.5], [14:40, 高台寺, 5.3], [15:20, 円山公園, 4.6], [16:20, 知恩院, 6.5], [17:30, 青蓮院門跡, 5.5]	27.3	5
モデルコース 2	[13:10, 八坂神社, 5.0], [13:30, 石塀小路, 5.1], [13:50, 京都霊山, 3.1], [15:20, 高台寺, 5.3], [16:00, 長楽寺, 3.4], [16:30, 円山公園, 4.9], [16:50, 知恩院, 7.0], [17:30, 青蓮院門跡, 5.5]	39.3	8

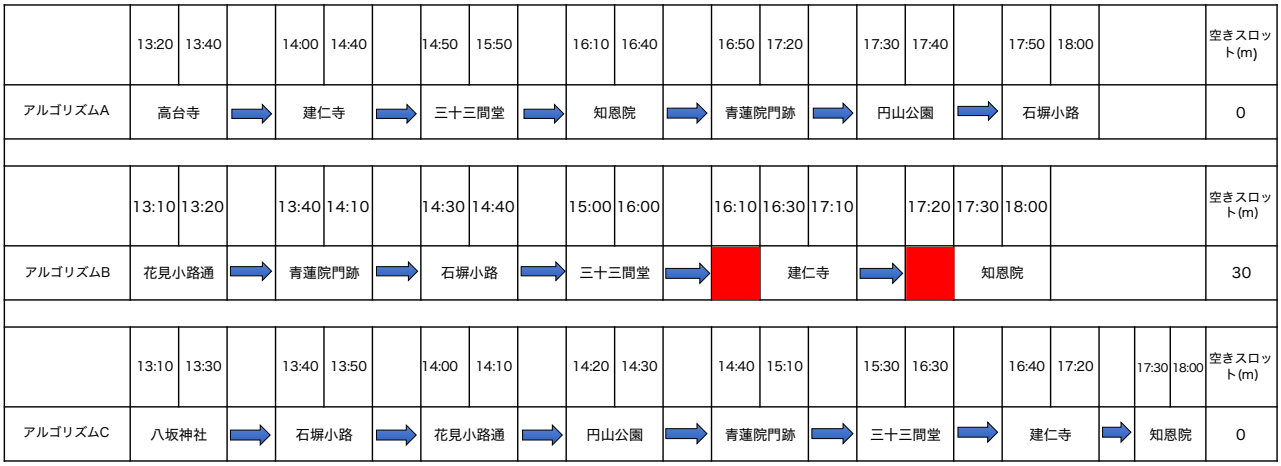

図 8: 各提案アルゴリズムの空きスロット

41.6，37.8，47.4）を比較した時，アルゴリズム A・C はモデルコース 2 より優れているが，アルゴリズム B はモデルコースより劣っていることがわかる．アルゴリズム B とモデルルート 2 の結果を比較したときに，知恩院に訪れているが，アルゴリズム B では 17 時 30 分に訪問することで，スコア値 8 を取得しており，モデルルート 2 では 16 時 50 分に訪問することで，スコア値 7 を取得している．知恩院により良い時間帯に訪問できているのは，アルゴリズム B である．これは青蓮院門跡でも同様のことが言える．全体のツアースコアを選択するか，各スポットの満足度を選択するかは，単純な優劣の関係ではないと考える．

7. 考察

本節では，表 3 の結果について評価および考察を行う．アルゴリズム A とアルゴリズム C を比較したときに，アルゴリズム C の方が満足度が大きい理由は，訪れる時間帯によってスポットのスコア値が異なるためである．アルゴリズム A では，行き当たりばったりに評価値が大きいスポットを訪問していくのに対して，アルゴリズム C では，評価値が上位 3 つの中で今後訪問できるスポットを考慮しながらスポットを選択していくため，アルゴリズム C ではより良い時間帯にスポットを訪問できている．アルゴリズム B が，他の提案アルゴリズムに比べてツアースコアが劣っている理由は，アルゴリズム B では合計空きスロットが他の提案アルゴリズムに比べて大きいためである（図 8）．赤四角は空き時間を示し，青矢印は各スポット間の移動を示す．図 8 では，各提案アルゴリズムの最大値ルートのみを示し，その最大値ルートのみ比較すると，アルゴリズム B での合計空きスロットが 30 分，アルゴリズム A・C は合計空きスロットが 0 分である．このことから合計空きスロットにより訪問スポット数及びツアースコアが，他の提案アルゴリズムに比べて劣っていることが考えられる．アルゴリズム C はアルゴリズム B に探索幅を考慮したアルゴリズムであり，出力解の空きスロットが 0 分であった．これは，探索幅を考慮することで，アルゴリズム B の問題点であるフラグメンテーションを解決することができているためだと考えられる．

8. おわりに

本稿では，観光経路探索問題に対し，静的スコア，動的スコア，事後期待スコアの 3 つの要素から成るツアースコアを定義し，本問題を実時間で解く時系列貪欲法，全体単一貪欲法，また，（実時間で解くことは現時点では難しいが，出力解のツアースコアが他の提案アルゴリズムに比べて良い）探索幅を考慮した全体貪欲法を提案した．次に，実際の京都府京都市東山区にある 20 箇所の POI に対して本アルゴリズムを適用し，出力解（ツアースコア）の質と求解までの計算時間の評価を行った．この評価実験では，*i*) 提案アルゴリズムの妥当性，*ii*) 計算時間に対して評価を行った．実験結果として，*a*) 時系列貪欲法・全体単一貪欲法は

現実的な計算時間でツアースコアが算出できる, *b*) 探索幅を考慮した全体貪欲法では，ツアースコアは最良解であったが，計算時間が非現実的である，*c*) 探索幅を考慮した全体貪欲法では，幅を大きくするにつれて，計算時間は指数関数的に増加するなどが分かった．また，実験結果として，3 つの提案アルゴリズムはトレードオフを考慮した準最適解を，それぞれ 1.9 ± 0.1 (s)，15.9 ± 0.5 (s)，7766.6 ± 8.9 (s) の計算時間で出力できることを確認した．今後はオンサイトで使用可能にするために，アルゴリズム C の改良による計算時間の短縮を目指す．改良案としては，現在は探索幅を考慮しているが，探索深さの制限を加えるかを考えている．また本評価では，一般向けのユーザに対してのスコア値 (静的スコア，動的スコア) の設定を行った．今後はスコア値（混雑度，天候情報など）をスケール変化させることで，多様なユーザ及び違うエリア（嵐山など）に対して本手法の有効性の検証を行う．

参考文献

[1] 国土交通省観光庁：旅行・観光消費動向調査，`https://www.mlit.go.jp/kankocho/siryou/toukei/shouhidoukou.html` (accessed 2020-05-16).

[2] Lamsfus, C., Alzua-Sorzabal, A., Martin, D., Salvador, Z. and Usandizaga, A.: Human-centric Ontology-based Context Modelling in Tourism., *KEOD*, pp. 424–434 (2009).

[3] Lim, K. H., Chan, J., Leckie, C. and Karunasekera, S.: Personalized trip recommendation for tourists based on user interests, points of interest visit durations and visit recency, *Knowledge and Information Systems*, Vol. 54, No. 2, pp. 375–406 (2018).

[4] Lim, K. H., Wang, X., Chan, J., Karunasekera, S., Leckie, C., Chen, Y., Tan, C. L., Gao, F. Q. and Wee, T. K.: PersTour: A Personalized Tour Recommendation and Planning System. (2016).

[5] Padia, P., Singhal, B. and Lim, K. H.: User-relative Personalized Tour Recommendation. (2019).

[6] Győrödi, R., Győrödi, C. and Derșidan, M.: An extended recommendation system using data mining implemented for smart phones, *International Journal of Computers & Technology*, Vol. 11, No. 3, pp. 2360–2372 (2013).

[7] Padia, P., Singhal, B. and Lim, K. H.: User-relative Personalized Tour Recommendation., *IUI Workshops* (2019).

[8] Matsuda, Y., Fedotov, D., Takahashi, Y., Arakawa, Y., Yasumoto, K. and Minker, W.: EmoTour: Estimating Emotion and Satisfaction of Users Based on Behavioral Cues and Audiovisual Data, *Sensors*, Vol. 18, No. 11 (online), DOI: 10.3390/s18113978 (2018).

[9] 中嶋勇人，新妻弘崇，太田　学：位置情報付きツイートを利用した観光ルート推薦，研究報告データベースシステム (DBS)， Vol. 2013, No. 28, pp. 1–6 (2013).

[10] Gao, H., Tang, J., Hu, X. and Liu, H.: Exploring temporal effects for location recommendation on location-based social networks, *Proceedings of the 7th ACM conference on Recommender systems*, pp. 93–100 (2013).

[11] Ye, M., Yin, P., Lee, W.-C. and Lee, D.-L.: Exploiting geographical influence for collaborative point-of-interest recommendation, *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 325–334 (2011).

[12] Logesh, R., Subramaniyaswamy, V. and Vijayakumar, V.: A personalised travel recommender system utilising social network profile and accurate GPS data, *Electronic Government, an International Journal*, Vol. 14, No. 1, pp. 90–113 (2018).

[13] Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M. and Steggles, P.: Towards a better understanding of context and context-awareness, pp. 304–307 (1999).

[14] 神嶌敏弘：推薦システムのアルゴリズム (1)，人工知能学会誌， Vol. 22, No. 6, pp. 826–837 (2007).

[15] 神嶌敏弘：推薦システムのアルゴリズム (2)，人工知能学会誌， Vol. 23, No. 1, pp. 89–103 (2008).

[16] Verma, R., Ghosh, S., Saketh, M., Ganguly, N., Mitra, B. and Chakraborty, S.: Comfride: a smartphone based system for comfortable public transport recommendation, pp. 181–189 (2018).

[17] Jevinger, A. and Persson, J. A.: Potentials of Context-Aware Travel Support during Unplanned Public Transport Disturbances (2019).

[18] 武　兵，村田佳洋，柴田直樹，安本慶一，伊藤　実：天気変化を考慮した観光スケジュール群の探索アルゴリズム，研究報告数理モデル化と問題解決 (MPS)， Vol. 2009, No. 3, pp. 1–6 (2009).

[19] 丸山敦史，柴田直樹，村田佳洋，安本慶一，伊藤　実：P-Tour：観光スケジュール作成支援とスケジュールに沿った経路案内を行うパーソナルナビゲーションシステム，情報処理学会論文誌，Vol. 45, No. 12, pp. 2678–2687 (2004).

[20] 平野陽大，諏訪博彦，安本慶一：ユーザのリソース消費を考慮した意思決定支援のための複数観光経路提示手法，*Multimedia, Distributed, Cooperative, and Mobile Symposium. (DICOMO 2018)*.

[21] JTB: 2019 秋限定の京都，JTB (2019).

[22] まっぷる：秋　紅葉の京都 2019，昭文社 (2019).