NAIST-IS-DD1061026

博士論文

コンテキストを考慮した観光スケジュールの立案手法

武 兵

2013年 2月 7日

奈良先端科学技術大学院大学
情報科学研究科 情報処理学専攻

本論文は奈良先端科学技術大学院大学情報科学研究科に
博士 (工学) 授与の要件として提出した博士論文である。

武 兵

審査委員：

伊藤 実 教授　　（主指導教員）
安本 慶一 教授　（副指導教員）
関　浩之 教授　（副指導教員）

コンテキストを考慮した観光スケジュールの立案手法*

武 兵

内容梗概

観光の振興は世界各地で地域活性化の重要な取組みとして期待されている．平成 23 年に訪日外国人旅行者数は 622 万人，日本人海外旅行者数は 1,699 万人に到達し，観光者へのサービス提供はますます重要になってきている．効率良く観光地を巡るには，巡回スケジュールを事前に作成し，観光地を対象とするナビゲーションが有効である．このようなサービスを提供するために，ユーザに対し目的地への経路案内や周辺情報の提供を行う様々なパーソナルナビゲーションシステムがすでに開発されているが，旅行者の満足度を大きく左右する天候や体力はまだ考慮されていない．例えば，天気が晴れの時と雨の時とでは，適した観光スケジュールが異なり，旅行者の満足度も当然異なる．また，同じ観光地でも，観光方式や観光時間によって必要な体力が異なり，場合によっては体力が足りなくなってしまうことがある．本論文では，天気，体力コンテキストをそれぞれ考慮した観光スケジュールの立案法を目指し，以下２つの手法を考案した．

１つ目の手法では，天気が確率的にしか予測できない場合を想定し，任意の天気変化パターンに対応した観光スケジュール群を算出する問題（以降，天候予測スケジューリング問題）を取り扱う．このスケジュール群は，出発地点を根として目的地ごとに分岐する木（スケジュール木と呼ぶ）で表現される．本問題の目的は，スケジュール木によって示された確率的なスケジュールの，ユーザ満足度期待値の総和を最大化することである．この問題は NP 困難であり，天候の変化パターンおよび変化タイミングの組み合わせは非常に多く，全てのパターンを対

* 奈良先端科学技術大学院大学 情報科学研究科 情報処理学専攻 博士論文, NAIST-IS-DD1061026, 2013 年 2 月 7 日.

応する場合，スケジュール木の規模が爆発的に成長する．実用時間で準最適解を得るために，欲張り法および局所探索法に基き，近似アルゴリズムを提案する．提案アルゴリズムは，まず欲張り法を用いて初期のスケジュール木を作成し，部分木を単位とした目的地の置換を繰り返し行うことにより，期待ユーザ満足度が高いスケジュール木を効率よく生成する．天候予測スケジューリング手法を評価するため，計算機シミュレーションにより，20 か所の観光地を取り扱うインスタンスをランダムに生成し，提案手法を適用した．その結果，本提案手法は，欲張り法のみを用いた場合と比べて平均 1.23 倍の期待値を持つスケジュール木を得ることができた．

2 つ目の手法では，観光中に休憩を適宜行うことで，体力の範囲内で，各目的地を好みの方式で観光するスケジュールを求める問題（以降，体力考慮スケジューリング問題）を取り扱う．本問題は，観光候補地を異なる方式で観光する場合の体力消費と得られる満足度をあらかじめ定義し，途中で体力を消耗しきることなく，スケジュール全体で得られる満足度をできるだけ大きくすることを目的とする．この問題も NP 困難であり，問題の規模が大きい時には，実用時間で最適解を算出することは困難である．実用時間で準最適解を得るため，ヒューリスティックな探索法である捕食法に基づいた手法を考案した．本手法では，まず，複数の観光スポットを廻る休憩なしのスケジュールを求め，局所探索を用いて適宜休憩を差し挟むことで解を求める．体力考慮スケジューリング手法を評価するため，異なる観光候補地数を有する複数のインスタンスを用いてシミュレーション実験を行った．その結果，候補地数 10 の場合，本提案手法は全探索で得られた解の 95.65%の満足度を有するスケジュールを 13 秒で得られることを確認した．

キーワード

ナビゲーションシステム，コンテキストアウェアネス，高度道路交通システム（ITS），広域・近傍探索，巡回スケジューリング，組合せ最適化アルゴリズム

Context-Aware Sightseeing Tour Scheduling Method *

Bing Wu

Abstract

Sightseeing promotion is expected as an important key to revitalize the local economy in various places of the world. The number of foreign tourists visiting Japan is 6,220,000, and the number of Japanese tourists traveling abroad is 16,990,000 in 2011, and so it is very important to provide the tourists a good service. There are several navigation systems which provide users sightseeing information and can compose a good schedule before sightseeing. However, they just use some simple contexts to calculate a schedule composed of multiple destinations. The user satisfaction in a sightseeing schedule depends greatly on weather and the stamina of tourists. For example, depending on the weather conditions such as rainy or sunny, the suitable sightseeing schedules are different. Moreover, in the same sightseeing spot, the required tourist's stamina differs among the sightseeing methods or sightseeing time, and the tourist's stamina may become insufficient during sightseeing. In this thesis, we tackle the problem to compose satisfactory sightseeing schedules for fickle weather and individually different stamina.

First, we formulate the problem of constructing a schedule of a sightseeing tour taking into account the probabilistic change of weather. In this scheduling problem, the schedule is represented as a $scheduling\ tree$ that consists of ordered sequences of visiting spots, where each spot is associated with user satisfaction degree which depends on weather. Then our problem is to find the scheduling tree that maximizes the expected value of total user satisfaction degree. Since this problem is NP-hard,

* Doctoral Dissertation, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD1061026, February 7, 2013.

there will be too many combinations between weather patterns and timings to plan the optimum scheduling tree in practice time. Since finding the optimum scheduling tree is computationally difficult, we propose an approximate algorithm based on the greedy search and the neighborhood search techniques. In order to evaluate the proposed method considering weather, we compared the performance of the algorithm with an existing method through randomly generated instances consisting of 20 spots. As a result, our proposed method constructed the scheduling tree whose expected value of total user satisfaction degree is 1.23 times higher than the existing method.

Second, assuming that the tourist's stamina and the travel time are restricted, we propose an algorithm for planning tour schedules where a favorite method (e.g., using a car or on foot) can be chosen for each sightseeing spot and rest can be taken properly during the travel time. We formulate the sightseeing scheduling problem which maximizes the user's satisfaction taking stamina into account. In this problem, rest times are allocated in schedules to satisfy stamina constraint. This problem is also NP-hard, and thus it is difficult to be solved in practical time. In order to obtain a semi-optimal solution in practical time, we propose a method that first derives a schedule visiting multiple sightseeing spots with no rest times based on a predatory search technique and then allocates rest times in the schedule using a local search technique. To evaluate the proposed method considering stamina, we compared this method with conventional methods through computer simulations for several different instances containing 10 sightseeing spots. As a result, the proposed method composed the schedule whose expected satisfaction is 95.65% of the optimum solution in 13 seconds.

Keywords:

Navigation system, Context-awareness, ITS(Intelligent Transport Systems), Extensive and Area-restricted Search, Travelscheduling, Combinatory optimization algorithm

目 次

図 目 次

表 目 次

1. 序論

観光は，旅行業，宿泊業，輸送業，飲食業，土産品業等極めて裾野の広い産業である．国の経済，人々の雇用，地域の活性化に大きな影響を及ぼすものであり，21 世紀のリーディング産業となる可能性を秘めている．平成 23 年の訪日外国人旅行者数は 621.9 万人，日本人海外旅行者数は 1,699 万人であった [1]．観光を通じて，見知らぬ地域や人々の暮らし，歴史・文化に触れることは，健康の維持に加え，創造力や柔軟な視点を養うことにつながり，世界の各地で地域活性化の鍵と考えられている．観光のための時間は往々にして限られているため，効率の良いスケジュールを立てることが重要である．

近年，ユーザに対し目的地への経路案内や周辺情報の提供を行う様々なパーソナルナビゲーションシステムが提案されている．パーソナルナビゲーションシステムの主な用途として観光への利用が挙げられる．エンタテインメントを目的としたものに，ユーザのコンテキスト（位置，時間，周辺環境など）に応じて動的に適切なルートを案内するシステム [2] や屋内施設でのルート案内 [3]，携帯端末に対し観光案内などの情報を提供するためのシステム [4] がある．また，旅行中すべての期間において案内を行うことを目的に，多様な旅行計画や，交通機関乗り換えに関する情報の提供，屋内外の案内のための統合アプリケーションが提案されている [5]．また，筆者の所属する研究グループでは，観光のためのパーソナルナビゲーションシステム P-Tour[6] を提案している．P-Tour は，ユーザの希望に応じた複数目的地の巡回スケジュールを立案し，また各目的地における案内機能を提供する．しかし，これらのシステムは，基本的に時間制約内で複数の目的地を巡回するスケジュールを計算するものであり，周辺のコンテキストを利用するが，旅行者の満足度を大きく左右する天候や旅行者自身の体力というコンテキストはこれまでに考慮されていない．本論文では，天気，体力といったコンテキストを考慮した二つの観光スケジュール作成支援に取組み，以下 2 つの手法を提案する．

1 つ目の手法では，観光における旅行者の満足度は，天気に大きく左右されるため，晴天時と雨天時とでは適したスケジュールは異なることが多いことに着目する．単純な対応としては，雨天時におけるユーザの満足度を目的地ごとのデー

タとして与える方法が考えられる．これにより，雨天時のスケジュールの立案が可能となる．しかし，天気が観光ツアーの間に移り変わる可能性がある場合には，天気の変化パターンおよびそのタイミングによって，適したスケジュールも変化する．従来手法では，このような場合のスケジュール群の立案は困難であった．

天候予測スケジューリング手法では，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する[7]．このスケジュール群は，出発地点を根とし天気に応じて分岐する目的地の列からなる木（スケジュール木と呼ぶ）で表現される．ユーザがある目的地を巡回した直後，そのときの天気に基づいて分岐先（つまり次に巡回する目的地）が決定される．各時点における天気は確率的に事前に与えられるとする．そのため，スケジュール木における分岐確率を計算することができる．得られたスケジュール木における根から葉に至る各パスは天気の変化の一つのパターンとそのパターンに適した複数目的地の訪問順序リストに対応しており，天気パターンの生起確率および目的地をその天気時に訪問した際の満足度を持っている．本問題の目的は，スケジュール木における満足度の期待値の総和を最大化することである．この問題はNP困難であり，天候の変化パターンおよび変化タイミングの組み合わせは非常に多く，全てのパターンを対応する場合，スケジュール木の規模が爆発的に成長する．実用時間で準最適解を得るため，欲張り法と局所探索法に基づいた近似アルゴリズムを新たに提案する．局所探索法（n-swap法と呼ぶ）は，スケジュール木の部分木内において目的地を交換するもので，同じ目的地を複数巡回するシーケンス（一つのスケジュール）を生成することなく，スケジュール群の局所探索を行うことができる．本スケジューリング手法により得られたスケジュール群を評価するため，シミュレーションにより従来手法との比較実験を行った．その結果，最も確率の高い天気と最も確率の低い天気を考慮して立てられたそれぞれのスケジュールに対し，天気が安定しない場合のインスタンスにおいて，7%から10%上回る期待値を持つスケジュールを得ることができた．

2つ目の手法では，観光旅行において体力が低下すると満足度が急激に下がるため，観光者の体力にあわせた無理のないスケジュール作成の必要性に着目する．観光者の体力を考慮した観光方式としては，登山ガイドブックに表示されるコー

スごとの必要体力の案内や，旅行会社による対象年齢別の旅行プランの推奨などがある．例えば，ケーブルカーで登山する場合，体力の消耗度が少なくなるが，得られる満足度も歩行登山の場合と異なる．体力を考慮した場合，いつどこで休憩を入れるかのスケジューリングが重要となる．体力と時間の制約条件の下で適宜休憩を入れ，満足度を最大化するような観光スケジューリングは複雑な最適化問題である．

体力考慮スケジューリング手法では，観光者の体力と時間を制約とし，適宜休憩を取りながら，複数の目的地に対し，好きな方式で巡るスケジュールの立案手法を提案する．本手法では，観光候補地を異なる方式で観光する場合の体力消費と得られる満足度をあらかじめ定義し，スケジュール全体で得られる満足度をできるだけ大きくすることを目的とする．観光候補地から選択する複数の目的地を廻る最も満足度の高いスケジュールを算出するためには，観光中に適宜休憩を挟むことで，体力残がマイナスにならないようにすることが有効である．この問題も NP 困難である．実用時間で準最適解を得るため，ヒューリスティックな探索法である捕食法に基づいた手法を考案した．本手法では，捕食法を用いて，休憩なしのスケジュールをまず求め，休憩を差し挟む回数及びタイミングを近傍探索で求めるというアプローチを採用する．体力考慮スケジューリング手法により得られた解の最適性を評価するため，10 観光候補地を有するインスタンスに対し提案手法と全探索で求めた解を比較した．その結果，最適解の 95.65%以上の満足度を持つ解を平均 12.99 秒で得ることができた．また，欲張り法と比較したところ，20 か所の観光候補地を持つインスタンスにおいて 1.36 倍の満足度を持つ解を得ることが出来た．

2. 研究背景

本章では，パーソナルナビゲーションシステムの関連研究について述べる．次に，本研究の先行研究である，観光のためのパーソナルナビゲーションシステム P-Tour について説明する．

2.1 関連研究

近年，ユーザに対し観光地の各スポット（以降，目的地と呼ぶ）への経路案内や周辺情報の提供を行う様々なパーソナルナビゲーションシステムが提案されている．パーソナルナビゲーションシステムに関する研究は，主にナビゲーションを目的とした経路探索手法に関するものである．エンタテインメントを目的としたパーソナルナビゲーションシステムとして，ユーザのコンテキスト（位置，時間，周辺環境など）に応じて動的に適切なルートを案内するシステム [2] や屋内施設でのルート案内 [3]，携帯端末に対し観光案内などの情報を提供するためのシステム [4, 8] が提案されている．文献 [5] では，旅行中すべての期間において案内を行うナビゲーションシステムが存在しないことに着目し，多様な旅行計画，交通機関乗換情報の提供，屋内外の案内のための統合アプリケーションの提案を行っている．また，関連するシステムとして，カーナビゲーションシステム上で動作するさまざまなアプリケーションを統合するシステム [9] がある．このシステムは，近隣の店の情報や事故の速報など，ユーザに情報を提供する様々なアプリケーションに対して優先順位をつけ，情報の表示やユーザの入力インターフェースを統合して提供する．

2 地点間の最適経路を求める手法として，文献 [10] で述べられているダイクストラ法や A* アルゴリズムが挙げられる．これらの手法はカーナビゲーションシステムなどに広く応用されている．しかし，これらは最適解を求める方法で，計算が完全に終了するまで解を得ることはできない．次に，複数地点を巡回する経路を求める問題として以下のものがある．文献 [11] では，複数の施設が存在するテーマパークへ多数の客が訪問するような状況において，各施設の利用・予約状況をもとにユーザへの情報提示や誘導をうまく調整することにより，全体の

混雑度を下げて客の満足度を上げる手法が提案されている．また，文献 [12] では，伊豆半島において，渋滞や交通事故の回避及び観光事業の活性化のために，PC や携帯端末または道の駅から，web を通じて，観光地や道路，公共交通機関，天気の情報をユーザに提供する “This Izu Navi” システムを開発している．文献 [13] では，駅周辺の特定領域のバリアフリープロジェクトの発展を目的とし，健常者のみならず車椅子，視覚障害の歩行者を対象として，様々な利用法を提供する “HOKO-NAVI” システムを開発している．実験を名古屋市栄区で行い，システムの有効性が示されている．

観光では，複数の観光地（目的地）を対象とし，中から満足度の高い経路をたどることが目的となる．観光問題は一般的にそれを巡回セールスマン問題と見なし，この問題を効率良く解くヒューリスティックアルゴリズムとして，Lin-Kernighan 法 [14]，焼きなまし法（Simulated Annealing）など，厳密に最適解を求めないことで計算時間を短くするアルゴリズムを用いる方法がある．より複雑な定義の問題を扱う解法としては，遺伝的アルゴリズム [15]，捕食法（Predatory Search: PS 法）[16, 17, 18]，大規模局所探索法 [19]，また Consultant-Guided 法 [20, 21] といった厳密解を保証しないヒューリスティックアルゴリズムを用いることも多い．

また，巡回路を求める問題として配送計画問題 (Vehicle Routing Problem, VRP) が広く研究されている [22, 23, 24, 25, 26, 27]．VRP は，様々な制約条件の下で，配送センターから，複数の配送車を用いて全ての客をちょうど一回ずつ訪問するような効率的な配送経路を求める問題である．文献 [22, 23, 24] では，VRP を解くための近似アルゴリズムを提案している．sweep 法 [22] では，客の座標を配送センターを中心とした極座標で表現し，角度的に近い客同士をグループ化することによって車両割り当てを決定する．その後，局所探索を用いて配送経路を求める．また，文献 [25, 26, 27] では，客への配送時間に制約条件（時間枠）がつくなどの拡張された VRP を扱っている．文献 [25] では，時間枠つき配送計画問題に，新たに移動時間に関する制約条件を加えている．またこの問題を解くために，動的計画法を用いて各客に対して適した配達時刻を決定することを局所探索法に組み込んだ解法を提案している．

2.2 観光のためのパーソナルナビゲーションシステム P-Tour

観光においては，ユーザの好む観光スポットをより多く回るスケジュールを立案することが望ましい．効率の良いスケジュール作成のためには，ナビゲーションシステムなどによるスケジュール作成支援が必須であると考えられる．筆者が所属する研究グループでは，複数の観光スポットを時間制約および重要度を考慮しつつ効率よく巡回するスケジュールの作成機能，および，作成したスケジュールに従って利用者に次の目的地へ経路案内する機能を持ったパーソナルナビゲーションシステム P-Tour[6] を提案している．

一般に，旅行者が観光スケジュールを作成する際には，できるだけ多くの目的地を与えられた時間内に効率よく回りたい，各目的地における施設の営業時間やイベントの開催時間，滞在時間などを考慮したい，といった要求が存在する．また，候補となる目的地が多数存在し，制限時間内に全てを訪れることが不可能な場合には，移動のコストや優先度を考慮し訪れる目的地の数を減らす必要がある．P-Tour は，ユーザが出発地と出発時刻，帰着地と帰着時刻，複数の観光候補地およびその重要度と時間制約（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ユーザの希望に添えるような巡回経路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．また P-Tour は決定したスケジュールに従い，観光中のユーザに経路誘導を行う機能を提供する．図 1 に示すように，P-Tour は，携帯端末上で実行されるクライアントモジュールとインターネットに接続された PC 上で実行されるサーバモジュールから構成される．なお，サーバから地図，目的地が納められたデータベースが参照できるものとする．P-Tour システムを利用し，ユーザは携帯電話や PDA 等を用いて，スケジュール算出やナビゲーションなどのサービスをネットワーク上のサーバから受けることができる．

観光中に天候が変化した場合においても，P-Tour はユーザの要求に応じてスケジュールを立案することができる．しかし，これは変化してからの対応であり，天候変化が確率的にわかっている場合において，あらかじめそれぞれの天候変化パターンに対応した複数のスケジュールを立てておくことはできなかった．本論文の 3 章では，天候変化の事前予測に基づいたスケジュール群の立案手法に取り

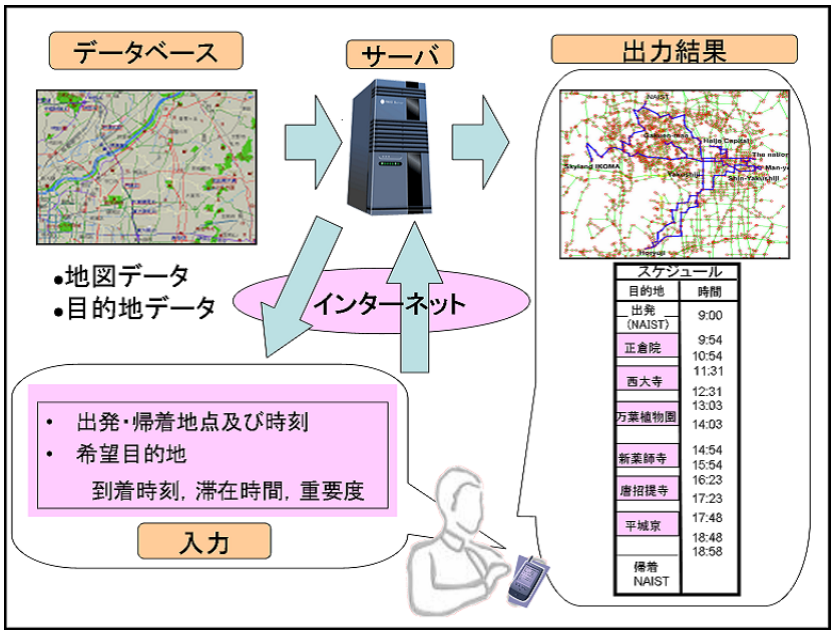

図 1 P-Tour の構成

組む．天気が確率的に予測できる場合を想定しており，任意の天気予報データに対応した観光スケジュール群を生成することが望まれる．つまり，天気の変化パターンおよびそのタイミングによって，適したスケジュールを求める．既存研究では，これを満たした研究は行われていない．

また，このシステムは，観光者の体力と観光方式に応じて目的地の満足度が変化する問題を取り扱っていない．観光中に，旅行者が食事と休憩を取ることに対して，どのタイミング，どこで取れば，最も体力を保てるかは組み合わせが多い最適化問題を含んでいる．そのため，4 章では，観光者の体力と時間を制約とし，適宜に休憩を取りながら，複数の観光地に対し，好きな方式で巡るスケジュールの立案手法に取り組む．

3. 天気の変化に対応可能な観光スケジュール群立案手法

3.1 概要

本章では，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の立案手法を提案する．このスケジュール群の評価は，対応する観光地を訪れた際の満足度と天気の確率を掛け合わせたものの総和（すなわち，期待満足度）により行う．つまり，生起確率の大きい天候変化パターンに対して適したスケジュールを含むほど期待満足度が高くなる．よって，天候が変化しやすい場合，また確実に天候が悪化する場合など，様々な天候変化パターンに対して確率的に適したスケジュール群を表現することが可能になる．期待満足度を最大化するスケジュール群の算出問題を解くため，欲張り法と局所探索法を用いた近似アルゴリズムを提案する．提案アルゴリズムは，まず欲張り法を用いて基となるスケジュール群を生成する．しかし，欲張り法では帰着経路などを考慮しないため，帰着経路が非常に長くなったら，帰着予定時間に間に合わないようなスケジュールが生成されやすい．そこで，局所探索法を用いて，欲張り法により得られたスケジュール群を改善する．局所探索法は，スケジュール木の部分木を単位とした目的地の置換を繰り返し行うことにより，期待ユーザ満足度が高いスケジュール木を生成する．本提案手法では，観光候補地 20 の場合，欲張り法の平均 1.23 倍の期待値を持つスケジュール木を得ることができた．

3.2 天気変化に対応する観光スケジューリング問題

本節では，天候を考慮したスケジュール立案において望まれる点について述べる．次に，天気の変化に対応可能な観光スケジュール木算出問題の定式化について述べる．問題の概要を例を用いて説明した後に，問題の定式化を与え，最後に本問題の NP 困難性について説明する．

3.2.1 天候変化に応じたスケジュール立案

天候による観光への影響: 観光地は，景観，歴史遺産，建物や施設，町並など様々な種類がある．しかし，屋外の観光地の多くは悪天候下における観光に適していない．雨具を身に着けたままの観光は不便であるし，雨天の展望台からは何も見えない．さらに山岳では，道を見失い遭難したり，崩落に巻き込まれる可能性すらある．以上のことから，観光時の天候に応じた観光地を選択し，巡回スケジュールを立案できることが望ましい．

利用できる天候情報: 最近は，かなり信頼できる天気予報を手軽に利用することが出来る．１日中晴天である場合，あるいは雨天であると予測される場合には，それぞれの天候において適している観光地のみを考慮すれば良い．しかし，天候が変化することがわかっている場合がある．例えば Yahoo!天気情報 [28] では，1日毎，6時間毎，3時間毎の天気予報を提供しており，事前に天候の回復・悪化を予測することができる．このような場合，観光中の天候変化を考慮した巡回スケジュールを立案できることが望ましい．

立案されるスケジュールの形態と計算量: 天気予報は確率的に与えられ，事前に天候変化の確実な時間を知ることは難しい．しかし，天候が悪化するかもしれないからといって，わずかでもその可能性がある間ずっと屋内の観光地を観光することは効率的ではない．従って，起こりうる天候変化パターンそれぞれに対してスケジュールを立案し，どのタイミングで天候が変化しても良いように備えておくことが望ましい．また天候変化は１回のみとは限らないため，これらパターンすべてに対するスケジュールを全探索で求めるためには非常に大きな計算量が必要であると考えられる．そこで，実用的な時間で近似解を算出する近似アルゴリズムが必要とされる．

3.2.2 問題概要

観光に適したスケジュールは，天候に応じて往々にして変化する（例：図 7）．旅行者は，旅行前にあらかじめ天気予報 web サイト [28] などを通じて，表 1 のような観光エリアの天気予報を入手できる．本問題では，事前に入手した天気予報情報に基づき，図 3 に示すようなスケジュール木を求める．この図では，奈良ホ

表 1 天気予報の例

	晴	曇	雨
09:00-10:00	80%	10%	10%
10:00-11:00	80%	10%	10%
11:00-12:00	10%	30%	60%
12:00-13:00	10%	20%	70%

図 2 天気ごとのスケジュールの例

テルを 9:00 に出発し，晴天であれば 10:00 に奈良公園を巡回することを示している．またその下に書かれた“（80 点, 20%）”は，奈良公園の晴天時における重要度が 80 点で，このときの巡回確率が 20%であることを示している．スケジュールごとの満足度は，含まれる目的地の重要度と，観光経路の短さ，帰着時間遵守に関する修正値の和として与えられる．そして満足度に巡回確率をかけることにより，スケジュール中における満足度の期待値が計算できる．本問題の目的は，スケジュール木全体の期待値の総和を最大化することである．

3.2.3 問題の定式化

3.1.3.1 入力

入力 I は，あらかじめ与えられたデータベース入力と，ユーザが探索時に与え

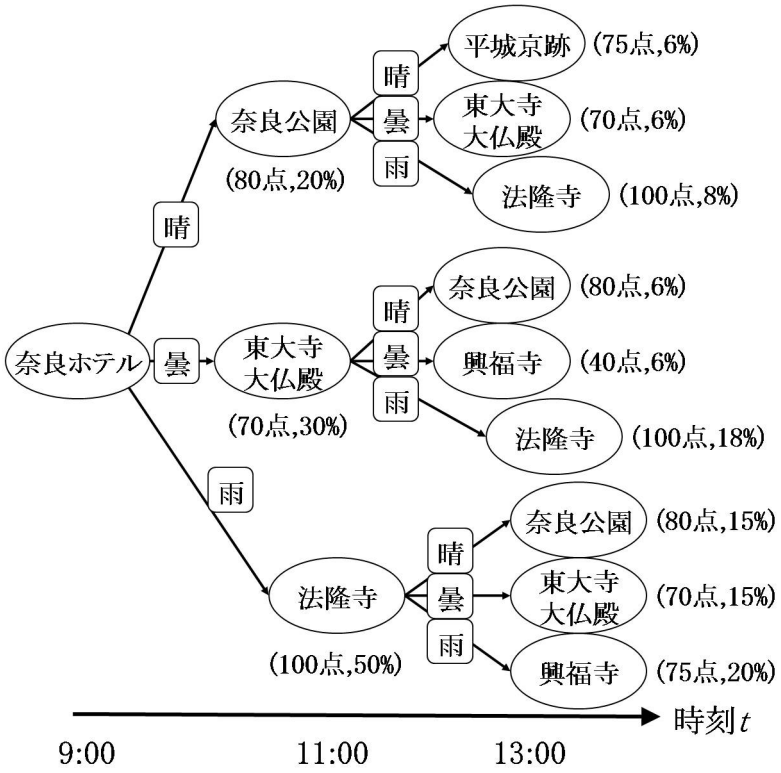

図 3 スケジュール木の例

るユーザ入力から成る.

<データベース入力> データベース入力として，道路地図および目的地のデータ，天気予報データが与えられる．データベース入力は以下の項から構成される.

- 道路地図データ: 道路網を示す有向グラフ $G = (U, E)$. 各エッジは以下の項目を持つ.
  - 頂点間距離: 道路網グラフ G のエッジはそれぞれ距離を持つ. 頂点 u_1, u_2 の間の距離は $dist(u_1, u_2)$ で与えられる.
- 目的地データ：観光スポットのデータであり，データベース上には次の項目を持つ.
  - $D = \{d_1, d_2, \ldots, d_n\}$: 観光スポットの集合（例：法隆寺，奈良公園など）を表す.

- $U = \{u_1, u_2, \ldots, u_n\}$: 観光スポットの位置の集合を表す．位置は道路地図データのいずれかの頂点に対応付けられている．

- 天気予報データ： 天気予報から導かれるデータであり，次の項目を持つ．

  - W: 天気の種類の集合．W = { 晴, 雨, 曇 } など．ここでは，観光において影響を与えうる天気すべてをあらかじめ入力する（例：降雪地域ならば雪など）．この天気はすべて排反である．w_1 が強風，w_2 が雨としたとき，w_1 かつ w_2 であるというような表現は許されない．よって風雨という天気が必要ならば，別に w_3 などとしてあらかじめ入力しておく必要がある．

  - $pw(w_i, t)$: 目的地が時刻 t において天気 w_i である確率を返す関数を表す．この値は天気予報により事前に入力として与える．気象庁などにより容易に手に入る天気予報の利用を想定し,「9:00–11:00 は晴天の確率が 80%，曇天の確率が 10%，雨天の確率が 10%」のように，観光が行われるすべての時間帯に対して与える（表 1）．

＜ユーザ入力＞ ユーザ入力として，滞在日数や出発・帰着時間，観光スポットの重要度，観光スポットにおける時間制約を与える．重要度は，ユーザがその観光スポットを訪れたい希望の度合いを示す値である．基本的に，巡回する観光スポットの重要度の合計が大きい経路が望ましい経路となる．また，この値は天気によって変わるため，すべての天気に対してそれぞれ与えられる．本システム運用時には，各目的地に対しデフォルトの重要度を設定することで，ユーザには特に訪れたい/ 訪れたくない目的地や観光中の装備により天気の影響を大きく受ける目的地の値のみを入力させることもできる．

ユーザ入力は以下の項から構成される．

- 旅程中の出発/到着データ:

  - $pd_s, pd_g \in D$: 出発/帰着地点を表す．

  - pt_s, pt_g: 出発時刻/帰着時刻，つまり出発/帰着地点における時間制約を現す．

- $speed$: ユーザの移動速度を表す.

- 目的地データ：観光スポットのデータであり，ユーザ入力として次の項目を持つ.

  - rst_i: 目的地 d_i の到着時刻に対する制約（例. “12:00 以前”）を表す.
  - dur_i: 目的地 d_i の滞在時間に対する制約（例. “到着時刻から 30 分”, “12:00 から 30 分” など）を表す.
  - $pre(d_i, w_j, t_i)$: 天気 w_j，時刻 t_i の時に，目的地 d_i に到着した場合の重要度を表す.
  - ϵ, ζ, η: 満足度の計算に使われる係数. それぞれ，目的地の重要度，観光経路の短さ（無駄な経路の少なさ），帰着時間遵守（時間オーバのペナルティ）をどれだけ重視するかを示す.

3.1.3.2 設計変数

本問題の設計変数はスケジュール木 $T_s = (U_s, E_s)$ である. 各ノードは目的地と天候を要素をとして持ち，ノード i の目的地と天候をそれぞれ d'_i, w'_i で示す. ここで，スケジュール木 T_s は根つき木である.

3.1.3.3 目的関数

本問題の目的は**期待値**を最大化するような根つき木 T_s を求めることである. 期待値を求める目的関数は後述の式 (2), (10) を用いて以下の式で与えられる.

$$exp(T_s) = \sum_{j \in L_s} pro(j) \cdot satt(j) \tag{1}$$

ここで，L_s はスケジュール木 T_s の葉ノードの集合である.

以降，式 (1) で用いる関数について述べる. 実際にはこれら関数は引数としてスケジュール木 T_s を必要とするが，記述の簡単化のために省略する. 本項で述べる以降すべての関数についても同様である.

$pro(i)$ はスケジュール木 T_s におけるノード i への到達確率を返す関数であり，以下の式で再帰的に与えられる.

$$pro(i) = \begin{cases} 1 & \text{if } i = root \\ pro(par(i)) \cdot pw(w'_i, dep(par(i))) & \text{otherwise} \end{cases} \tag{2}$$

ここで，root はスケジュール木 T_s の根ノード，$par(i)$ はスケジュール木 T_s におけるノード i の親ノードを返す関数，w'_i は $par(i)$ から i に至るリンクに付与された天候である．$pw(w,t)$ は天気予報データの項で定義した，時刻 t で天気が w である確率である．$dep(i)$ はノード i の出発時間を返す関数であり，以下の式で再帰的に与えられる．

$$dep(i) = \begin{cases} pt_s & \text{if } i = root \\ dep(par(i)) + dur'_i + dist(d'_i, d'_{par(i)})/speed & \text{otherwise} \end{cases} \tag{3}$$

$satt(j)$ は，スケジュール木 T_s において根ノード $root$ から葉ノード j に到達するスケジュール（以降，これをスケジュール S_j と呼ぶ）の満足度を与える関数で，以下の式で与えられる．

$$satt(j) = \epsilon \cdot desp(S_j) + \zeta \cdot disp(S_j) + \eta \cdot timp(S_j) \tag{4}$$

ここで ϵ, ζ, η は目的地の重要度，観光経路の短さ，帰着時間遵守のどれを重要視するかを調整する重み係数である..

$desp(S_j)$ は，スケジュール S_j において制約条件を満たした目的地における重要度の合計を返す関数で，以下の式で与えられる．

$$desp(S_j) = \sum_{i \in nod(S_j)} satd(i) \tag{5}$$

$$satd(i) = \begin{cases} pre(d'_i, w'_i, tima(i)) & \text{if } tima(i) \leq rst_i \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

ここで，$nod(S_j)$ はスケジュール S_j 内のノードから根ノード root を除いたノード集合を返す関数である．$d'_i \in D$ はノード i における目的地，$w'_i \in W$ はノード i

における天候である．$pre(d, w, t)$ は目的地データの項で定義した．天気 w 時刻 t に，目的地 d に到着した場合の重要度である．

$tima(i)$ は，スケジュール木 T_s においてノード i の目的地に到着する時間を返す関数である．

$$tima(i) = \begin{cases} pt_s & \text{if } i = root \\ dep(par(i)) + dist(d'_i, d'_{par(i)})/speed & \text{otherwise} \end{cases} \tag{7}$$

$disp(S_j)$ は，スケジュール S_j の経路長さによるペナルティ値（負の数）を返す関数で，以下の式で与えられる．

$$disp(S_j) = -\sum_{i \in nod(S_j)} (dist(i, par(i))) \tag{8}$$

$timp(S_j)$ は，スケジュール S_j において帰着時間を超過したときのペナルティ値（負の数）を返す関数で，以下の式で与えられる．

$$timp(S_j) = \begin{cases} 0 & \text{if } tima(j) \leq pt_g \\ tima(j) - pt_g & \text{otherwise} \end{cases} \tag{9}$$

3.1.3.4 制約条件

スケジュール木 T_s は以下の制約条件を満たしていなければならない．

- スケジュール木は $|W|$ 分木であること
  - つまり，各スケジュールは冒頭部分を必ず共有する．
  - 完全 $|W|$ 分木である必要はなく，根ノードから葉ノードまでの深さは一定でなくて良い．
- 葉ノードを除く各ノードは，W 内のすべての天候に対応する子ノードを持つこと．
- 根ノードの目的地 $d_{\text{root}} = pd_s$ であること．
- 葉ノードの目的地 $d_j = pd_g, j \in L_s$ であること．

- 各スケジュール（根から葉までの目的地の列）の目的地集合は，同じ目的地を複数個含まないこと[1]

3.2.4 NP 困難性

天気の変化に対応可能な観光スケジュール群算出問題は，観光時間の間ずっと天気が同一である場合，天気を考慮しない場合の観光スケジュール立案問題 [7] に帰着できる（例えば一日中晴れの場合）．天気を考慮しない場合の観光スケジュール立案問題は，出発地点と帰着地点を同一にして，各スポットの時間制約を削除し，最適解において必ずすべてのスポットを巡回するような重要度を与えることにより，NP 困難であることが知られている巡回セールスマン問題に帰着できる．以上により，天気の変化に対応可能な観光スケジュール群算出問題は NP 困難である．従って，実用的な時間で最適解を求めることは困難である．そのため以下では，欲張り法と局所探索法を用いた近似アルゴリズムを提案する．

3.3 天気変化に対応可能な観光スケジューリングアルゴリズム

提案する近似アルゴリズムは (1) 欲張り法，(2) 帰着スケジュール生成，(3) 局所探索法の 3 つの部分からなる．

1. 入力に欲張り法を適用し，初期のスケジュール木を生成する部分．ここでは帰着時刻を考慮せず，完全木を生成する．
2. 帰着時刻を考慮しないスケジュール木から，帰着時刻までに帰着地点に帰るスケジュール木を生成する部分．
3. 基となるスケジュール木に局所探索を適用し，改善する部分．

本節では，これらの部分について詳しく述べる．

[1] スケジュール木内の異なるスケジュールにおいては，同じ目的地を複数個含んでいて良い．

3.3.1 欲張り法

この部分では，欲張り法を用いて場当たり的にスケジュール木を構成する．スケジュール木の各ノードからの天気の分岐先のノードそれぞれに対し，最も満足度が高いと予測される目的地を割り当てる．提案する近似アルゴリズムでは，スケジュールにおける木構造をヒープ構造で表す.

この部分では，高さ $height$ の完全木を生成する．ここで $height$ は生成する完全木の高さであり，スケジュール木の各パスが含みうる最大の目的地数とする．この値は，出発時刻と帰着時刻の差，および目的地における最小滞在時間から計算することができる．

1. 帰着地点に到達していない葉ノードを格納しておく集合（以下，探索中節点集合）を表す変数 S を用意し，空集合で初期化する．
2. 根の節点を考える．この節点に，目的地として出発地点 pd_s を，この節点を出発する時刻として，最初の出発時刻 pt_s を割り当てる．この節点を探索中節点集合に加える．
3. 探索中節点集合 S が空であれば，アルゴリズムを終了する．そうでなければ，ここから任意の節点を一つ取り出す．この節点を注目節点とする．
4. 注目節点に割り当てられた目的地から，満足度を最大にするような次の目的地を欲張り法を用いて選択する．満足度は天気によって異なるため，天気ごとに１つずつ選ぶ．注目節点の子節点にこれら目的地を割り当て，次の目的地における到着時間と出発時間を計算する．子節点の深さが $height$ よりも小さければ，これら子節点を探索中節点集合 S に加える．注目節点を S から取り除く．
5. ステップ (3) へ戻る．

3.3.2 帰着スケジュール生成

3.3.1 項で述べたアルゴリズムは，帰着地点に帰ってくるスケジュールを必ずしも生成しない．そこで，本項で述べるアルゴリズムを用いて，これを帰着時刻

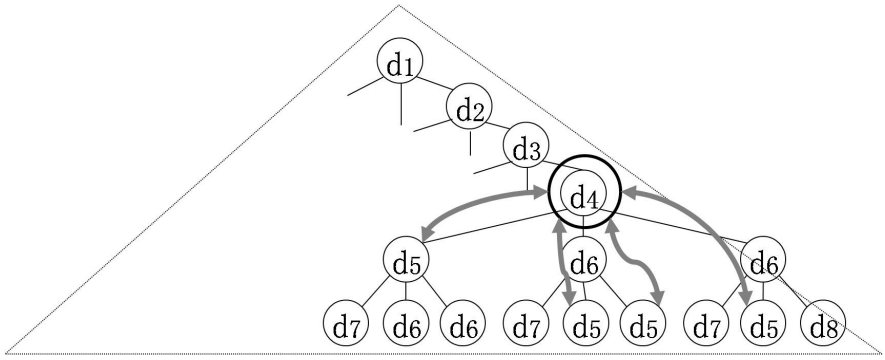

図 4 n-swap 法の例

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．以下に，帰着スケジュール生成アルゴリズムを示す．

1. 探索中節点集合 S を空にする．スケジュール木の葉節点すべてを探索中節点集合 S に加える．
2. 探索中節点集合 S が空であれば，アルゴリズムを終了する．そうでなければ，ここから任意の節点を一つ取り出す．この節点を注目節点とする．
3. 注目節点に割り当てられた目的地から，直接帰着地点に帰った場合，帰着時刻までに帰ることができるかどうかを計算する．これは，目的地と帰着地点の間の距離，およびその節点の出発時刻から計算できる．帰ることができるなら，ステップ (2) に戻る．
4. 注目節点に，目的地として帰着地点を割り当て直す．その親節点の出発時刻および割り当てられた目的地から，到着時刻を計算する．親節点を探索中節点集合 S に加え，ステップ (2) に戻る．

3.3.3 局所探索法

本項では，スケジュール木を改善する局所探索法について述べる．この手法では，まずスケジュール木内のランダムに選ばれた二つの節点に割り当てられた目的地を交換する．スケジュール木の各パスは同じ目的地を複数回巡回するパター

ンを含んではならないため，この交換は部分木を対象として行う．この交換を複数回行った後のスケジュール群の期待値を計算し，改善されていればそれを採用，悪化していれば破棄する．この繰り返しの数を**交換回数** n とし，この局所探索法を n-swap 法と呼ぶ．この過程を規定回数繰り返し，アルゴリズムは終了する．

1. スケジュール木の中の節点をランダムに選択する．この節点を交換元節点とする．また，この節点に割り当てられている目的地を交換元目的地とする．図 4 の例では，黒丸で囲んだ節点を選択しており，交換元目的地は d_4 である．スケジュール木は天気ごとに分岐するため，同じ目的地が複数の節点に割り当てられている場合があることに注意が必要である．
2. 全ての目的地の集合から交換元目的地 (d_4) と交換元節点の先祖の目的地 (d_1, d_2, d_3) 以外の目的地をランダムに 1 つ選択する．選択した目的地を交換先目的地とする．図 4 の例では，目的地 d_5 を選択している．
3. 交換元節点を根とする部分木内において，交換先目的地を割り当てられているすべての節点の目的地に対して，交換元目的地 (図 4 では目的地 d_4) を上書きする．その後で，交換元節点に交換先目的地 (図 4 では目的地 d_5) を上書きする．部分木内に交換先目的地を持つ節点が一つもない場合もある．その場合，交換元節点にのみ目的地の上書きを行う．
4. 交換回数が n 未満であれば，ステップ (1) に戻る．
5. 得られたスケジュール木に 3.3.2 項のアルゴリズムを適用し，修正された木の期待値を計算する．期待値が悪化していた場合，元に戻す．
6. 繰り返しが繰り返す回数に達すれば終了する．そうでなければ，ステップ (1) に戻る．

3.4 評価実験

本節では，天候予測スケジューリング手法の有効性を評価するために行った実験について述べる．以下の 3 つの実験を行った．(1) 天候予測スケジューリング手法の n-swap 法の効果を評価するため，交換回数 n の値を変えて得られたスケ

表 2 目的地属性による重要度係数

天気	屋外型	室内型	その他
晴	1.0	0.4	0.8
曇	0.6	0.5	0.8
雨	0.2	1.0	0.8

ジュール木の期待値を計測した．(2) 天候予測スケジューリング手法により得られた解の品質を評価するため，従来手法により得られた解と比較した．(3) 天候予測スケジューリング手法の有効性を評価するため，本提案手法と全探索法と比較して解の最適性を評価した．

実験環境として，CPU Celeron 2.00GHz，メモリ 2.00GB，OS WindowsXP Professional の PC を用いた．プログラミング言語として Java のバージョン 2.3.2 を用いた．

実験 (1) と (2) には，ランダムに生成された 10 個のインスタンスを用いた．これらインスタンスは，13500m × 9000m のフィールドの中にランダムに配置された 20 個の観光スポットを含んでいる．観光スポット間の距離として，観光スポット間の直線距離を与えた．各観光スポットの重要度は天気ごとに 50-100 の間の値の一様乱数で与えた．また滞在時間は，1, 1.5, 2, 2.5, 3 時間のいずれかの値をランダムに与えた．ランダム生成された 10 個のインスタンスの詳細は付録 A に後述する．また実験 (3) に対しては，奈良をモデルとした 15 個の観光スポットを含むインスタンスを用いた．実験 (3) の各観光スポットの重要度は，表 2 に示す重要度係数を掛けることにより，天気に応じた重要度としている．例えば，国立国会図書館 （基本重要度 50，室内型）において曇（室内では係数 0.5）であれば重要度は $50 \times 0.5 = 25$ 点となる．自動車での移動を想定し，すべての実験において，目的地間の移動速度を 40km/h とした．満足度の係数として，$\epsilon = 1, \zeta = 1, \eta = 0.5$ とした．つまり，移動距離 1km ごとに満足度は 1 点減点され，1 分遅延するごとに 0.5 点減点される．

表 3 天気予報データのリスト

天気変化	5 時間観光	8 時間観光	晴れ	曇り	雨
晴れ－晴れ	09:00-12:00	09:00-13:00	80%	10%	10%
	12:00-15:00	13:00-18:00	80%	10%	10%
雨－雨	09:00-12:00	19:00-13:00	15%	15%	70%
	12:00-15:00	13:00-18:00	15%	15%	70%
晴れ－雨	09:00-12:00	09:00-13:00	80%	10%	10%
	12:00-15:00	13:00-18:00	10%	10%	80%
雨－晴れ	09:00-12:00	09:00-13:00	15%	15%	70%
	12:00-15:00	13:00-18:00	70%	15%	15%
不安定	09:00-12:00	09:00-13:00	33%	33%	33%
	12:00-15:00	13:00-18:00	33%	33%	33%

3.4.1 局所探索交換回数 n の評価

天気の変化に対応可能な観光スケジュール木算出問題は NP 困難であり，多くの局所解を持っている．天候予測スケジューリング手法は局所探索法を用いているが，良い解を求めるには，どれだけうまく悪い局所解から脱出できるかが重要となる．n-swap 法の n の値は局所探索時における解空間上の移動距離であり，n の値が大きければ局所解からの脱出能力は向上するが，収束し難くなると考えられる．また，n-swap 自体を繰り返す回数が多いほど良い解は求められるが，計算時間は増加する．そのため，適切な n の値と繰り返し回数を求めることが重要である．そこで，適切な n の値と繰り返し回数を評価するため，欲張り法により得られたスケジュール木の期待値と，さらに n-swap 法を適用した後の期待値を計測し，これらを比較した．この値を改善率（n-swap 法後期待値/欲張り法後期待値）とする．

ランダムに生成された地図データ 10 個に対し，表 3 に示す 5 つの天気予報データ（晴→晴，晴→雨，雨→晴，雨→雨，安定しない）を用いた．n の値を変えて得られたスケジュール群の期待値の改善率と計算時間を，一定の繰り返し回数ご

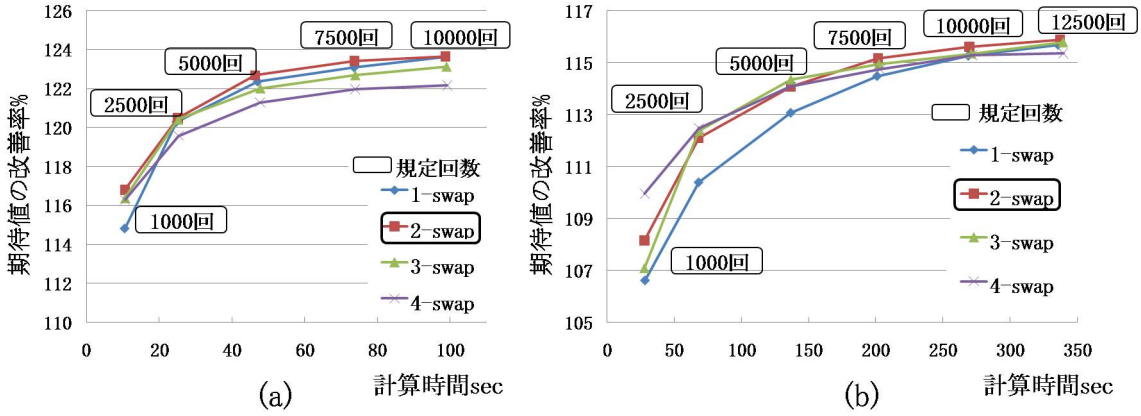

図 5 n-swap 法の機能評価-5 時間 (a) と 8 時間 (b)

とに計測し，これらを評価した．全体の観光時間の長さに応じて，解が収束するために必要な繰り返し回数は異なってくると考えられる．そこで観光時間が 5 時間の場合（出発時間 9 時，帰着時間 14 時）と，8 時間の場合（出発時間 9 時，帰着時間 17 時）について実験を行った．n の値は 1 から 4 とした．表 3 に示す 5 つの天気予報データ，各種類の繰り返し回数に対して，各 n-swap 法の改善率と計算時間を 5 試行の平均として計算した．

観光時間が 5 時間で，異なる地図データ 5 個（地図 1，2，3，4，5）を用いた場合の結果を図 5(a) に示す．この場合，繰り返し回数が 5000 回を超えたあたりで解は収束した．一方，観光時間が 8 時間で，異なる地図データ 5 個（地図 6，7，8，9，10）を用いた場合の結果を図 5(b) に示す．この場合，繰り返し回数が 7500 回を超えたあたりで解は収束した．提案手法では，一回全探索木を構築した後，n 回の swap を行い，swap 後の探索木から準最適解を探索する．観光時間が 5 時間（地図 1〜5）では，この操作が 125 回繰り返されると約 1 秒となる．観光時間が 8 時間（地図 6〜10）では，この操作が 35 回繰り返されると約 1 秒となる．全探索木の構築と解の探索が swap 操作よりもはるかに時間がかかるため，n の値による演算時間への影響は認められていない．図 5 より，観光時間が 5 時間，8 時間のいずれの場合でも，n の値にかかわらず，欲張り法のみを用いた場合と比較して n-swap 法を用いた場合は期待値を改善することができた．このことから，欲張り

法に加えて n-swap 法を適用する手法は有効であると考えられる．また，$n = 2$ の場合が良い性能を示していることが分かる．n-swap 法はランダムに解を変化させた後に，解が改善されていれば採用する発見的手法である．n はランダムに解を変化させる近傍距離を示している．局所解に陥らない限り，n の値は小さいほうが，細かい改善を繰り返すことができる．逆に n が大きくなると，大域的な探索能力が向上する．しかし解の一部を改善するが，もう一部では悪化させるようなことが多くなり，局所的な改善効率は悪くなると考えられる．観光時間が 5 時間から 8 時間になると，収束するまでの平均計算時間は 3.5 倍程度に増加した．これは，スケジュール木の高さが増加するにつれ，木の節点数が大きく増加するためである．

3.4.2 従来手法との比較評価

天候予測スケジューリング手法によりスケジュール群を立案することの有効性を調べるため，スケジュールを一つだけ立案するような従来手法との比較実験を行った．従来手法として P-Tour[6] を用いた．これは天気変化に対応するスケジュールを立案するために開発されたものではない．しかし，スケジュールの満足度を計算する際に，各時刻における天気に応じた重要度を用いることにより，時刻ごとの天気が明確である場合（つまり，100%予測できる場合）のスケジュールを立案することができる．実験では，それぞれの時刻において，最大（または最小）の確率を持つ天気を与えた．

また，順回路を求める問題である VRP に着目し，もう一つの巡回路を求める従来手法として sweep 法と呼ぶ比較手法を作成した．sweep 法では，まず出発地点を配送センター，目的地を需要者とみなし，観光スケジュール木算出問題を VRP 問題と見立てて配送経路を決定する．この部分に文献 [22] の手法を用いる．そして 1 台の車の経路を 1 つのスケジュールとみなし，2.2.3 項で述べた目的関数を用いてそれぞれのスケジュールの満足度を計算する．最大の満足度を持つスケジュールを文献 [22] の 2-opt 法で改善し，最終的なスケジュールを得ることとした．

3.3.2.1 比較方法

P-Tour や sweep 法で得られたスケジュールは木構造を持っていないため，その

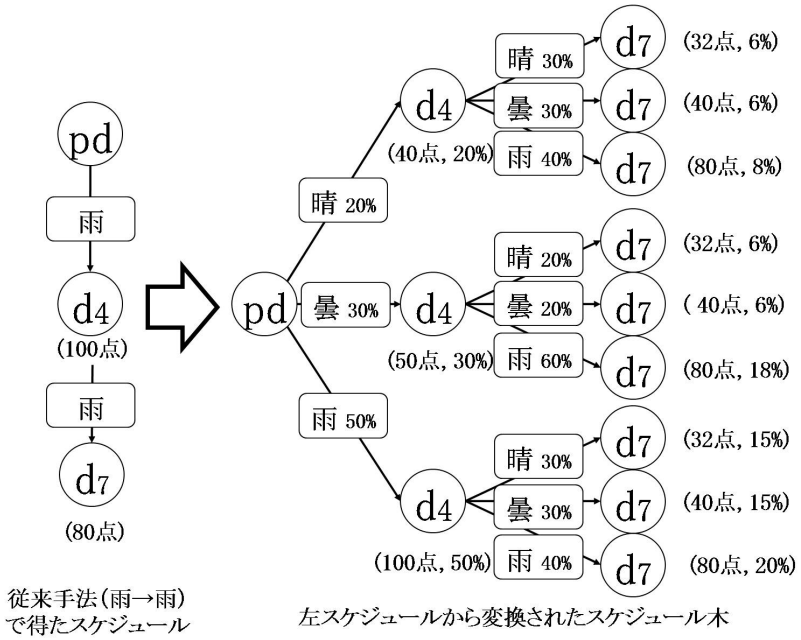

図 6 スケジュールからスケジュール木への変換

ままでは天候予測スケジューリング手法により得られたスケジュール群と比較することができない．そこで従来手法により得られたスケジュールを，どの天気においても巡回する目的地を変更しないようなスケジュール木に変換する．図 6 では．$pd \to d_4 \to d_7$ と目的地を巡回するスケジュールは，どの天気予報データにおいても $pd \to d_4 \to d_7$ と目的地を巡回するようなスケジュール群に変換される．この変換により得られたスケジュール木と，天候予測スケジューリング手法により得られたものを以下の項目において比較した．

- **期待値:** 式 (1) により計算される期待値．これは，問題の目的関数値そのものである．
- **満足度:** 式 (4) により得られるそれぞれのスケジュールの満足度．生起確率が最大であるスケジュールの満足度（**満足度（最大天候）**）と生起確率が最小であるスケジュールの満足度（**満足度（最小天候）**）を用いて比較した．

図 6 では，満足度（最大天候）に対応するスケジュールは雨→雨のものであり，

表 4 スケジュール木期待値改善率の比較結果

改善率=天候予測スケジューリング手法の値/P-Tour，sweep 法の値 × 100%			
実験データ		P-Tour からの改善率	sweep 法からの改善率
地図	天気変化	期待値	期待値
地図データ 1	晴れ－晴れ	102.97%	102.97%
地図データ 2		109.11%	109.11%
地図データ 3	雨－雨	100.77%	100.45%
地図データ 4		100.21%	103.45%
地図データ 5	晴れ－雨	102.58%	100.75%
地図データ 6		100.97%	100.97%
地図データ 7	雨－晴れ	101.13%	101.13%
地図データ 8		105.06%	105.06%
地図データ 9	不安定	107.62%	106.82%
地図データ 10		110.26%	110.26%
平均値		104.06%	104.09%

そのときの生起確率は 20%である．一方，生起確率が最小であるスケジュールは，図 6 において複数ある．このような場合は，集合 W における並び順（ここでは晴→曇→雨の順）の早いほうを選ぶこととした．よって，満足度（最小天候）に対応するスケジュールは晴→晴のときのスケジュール（生起確率 6%）である．改善率は，天候予測スケジューリング手法による期待値や満足度の改善度合いを示す指標であり，“ 天候予測スケジューリング手法の値/従来手法法の値 × 100% ” で与えられる．つまり，改善率が 110%であれば 1 割の改善，80%であれば 2 割の悪化を示す．

3.3.2.2 比較結果

実験には，ランダムに生成された地図データと重要度の組み合わせ 10 個を用いた．ここではそれぞれの地図データに対して，天気予報データを一つずつだけ与

表 5 最大，最小天候の場合にスケジュール満足度改善率の比較結果

改善率=天候予測スケジューリング手法の値/P-Tour，sweep 法の値 × 100%					
実験データ		P-Tour からの改善率		sweep 法からの改善率	
地図	天気変化	満足度		満足度	
		最大天候	最小天候	最大天候	最小天候
地図データ 1	晴れ－晴れ	97.27%	196.65%	97.27%	196.65%
地図データ 2		92.57%	562.24%	92.57%	562.24%
地図データ 3	雨－雨	94.93%	119.04%	94.63%	118.53%
地図データ 4		95.79%	110.69%	95.54%	114.84%
地図データ 5	晴れ－雨	99.09%	233.03%	97.69%	224.26%
地図データ 6		94.67%	160.09%	94.67%	160.09%
地図データ 7	雨－晴れ	99.52%	137.03%	99.52%	137.03%
地図データ 8		98.72%	157.28%	98.72%	157.28%
地図データ 9	不安定	82.17%	184.74%	81.75%	182.29%
地図データ 10		90.16%	181.29%	90.16%	181.29%
平均値		94.48%	204.21%	94.25%	203.45%

えている．出発時間を 9 時，帰着時間を 17 時とした．これら地図データに対して P-Tour，sweep 法，天候予測スケジューリング手法を適用して期待値を計測した．実験結果を表 4 に示す．これらは 10 試行の平均である．天候予測スケジューリング手法は，P-Tour に対して平均 104.06%，sweep 法に対して平均 104.09%，の改善率を示した．また，天候が安定しない場合においては，106.87%から 110.26%という高い改善率を示した．天気が確定的にわかっている場合でなければ，本提案手法によりそれぞれの天気に対するスケジュールを用意しておくことが有効であり，特に天候が不安定なときにそれが顕著であることを示している．

スケジュールの満足度に関する比較結果を表 5 に示す．満足度（最大天候）で比較した場合，天候予測スケジューリング手法はいずれの場合においても 100%未満

表 6 観光スポットリスト

NO	観光スポット名	基本重要度	滞在時間	属性
1	東大寺大仏殿	91	2h	その他
2	スカイランドいこま	39	2.5h	屋外型
3	平城京跡	78	1.5h	屋外型
4	春日大社	74	1.5h	屋外型
5	猿沢池	41	1.5h	屋外型
6	国立国会図書館	50	2.5h	室内型
7	奈良国立博物館	66	2h	室内型
8	郡山城跡	40	1.5h	屋外型
9	興福寺	79	1.5h	室内型
10	奈良公園	83	2h	屋外型
11	法起寺	53	1h	室内型
12	帯解寺	33	1h	室内型
13	法隆寺	100	2h	室内型
14	石上神宮	51	1h	その他
15	弘仁寺	34	1h	室内型

の数値を示している．しかし，満足度（最小天候）で比較した場合，最低で 110%，多ければ 562%と，非常に大きい改善率を示している．これらのことから，天候予測スケジューリング手法では生起確率が比較的低いスケジュールにおいての満足度の改善が顕著であり，このことがスケジュール木全体の期待値の改善につながっていると考えられる．

3.4.3 奈良県地図を用いた最適性評価

本項では，ランダム生成でなく，実際の奈良市内における観光地図をインスタンスとして用いた 3 つの実験について述べる．(1) 奈良市内の観光スポットを対象とし，アンケートを用いて観光スポットの重要度を設定した．(2) 設定された

表 7 全探索法との比較結果

方法	計算時間	期待値
全探索法	18.3 hours	131.61
欲張り法	18 msec	123.09
2-swap 法	3756 msec	131.61

重要度および国土地理院発行の数値地図 50000 を用いて，天候予測スケジューリング手法を全探索法と比較して解の最適性を評価した．(3) 同様の重要度および地図を用いて，天候予測スケジューリング手法を欲張り法と比較して解の質を評価した．

3.3.3.1 アンケートによる重要度設定

観光スポットの重要度は，旅行者から見た主観的な値である．その主観的な値がどのような値となり，どのように分布するかによって，観光地図の性質は異なってくる．また，観光スポットの数が多いため，旅行者が旅行のたびにそれらの値をすべて入力することは繁雑である．そこで，奈良市内の地図を用い，奈良在住の観光経験者 10 名にアンケートを取ることにより，各観光スポットの重要度，滞在時間を決定した．観光スポットの数は 15 である．結果を表 6 に示す．重要度は 10 名の与えた値の平均値，滞在時間は最も長かった時間を用いている．

3.3.3.2 最適性の評価

天候予測スケジューリング手法により得られた解の最適性を評価するため，全探索により得られた解との比較実験を行った．また，欲張り法部分だけで求めた解とも比較した．天候予測スケジューリング手法の局所探索部分には，2-swap 法を用いた．インスタンスとして，表 6 の幾つかの観光スポットを訪問するインスタンスを用いた．表 1 に示した天気予報データを用いて，出発時間を 9 時，帰着時間を 12 時とした．

実験結果を表 7 に示す．これは 10 試行の平均値である．2-swap 法は，全探索により 18.3 時間かかるようなインスタンスに対して 3756msec で最適解を求めることができた．一方，欲張り法部分のみを用いた場合は，18 msec で解が求まった．しかし，欲張り法ではこの小さいインスタンスに対しても最適解を求めるこ

表 8 欲張り法との比較結果

天気変化	方法	計算時間	期待値
晴れ - 晴れ	欲張り法	253 msec.	301.62
	2-swap 法	55.16 sec.	316.33
晴れ - 雨	欲張り法	258 msec.	289.71
	2-swap 法	53.79 sec.	332.25
不安定	欲張り法	261 msec.	264.69
	2-swap 法	55.49 sec.	286.89

とができなかった．これらのことから，天候予測スケジューリング手法は十分高速で効果的であることが分かった．

3.3.3.3 2-swap 法部分の評価

天候予測スケジューリング手法の局所探索法部分を評価するため，欲張り法部分のみを用いた場合と，本提案手法全体を用いた場合において得られたスケジュール木の期待値を比較した．インスタンスとして，観光スポット数 15 を持つ (表 6 に示す) 奈良県の地図を用いた．出発時間を 9 時，帰着時間を 17 時とした．2-swap 法を適用した．天気変化パターンとしては晴→晴，晴→雨，終日安定しない (図 3 に示す) 場合の 3 つのパターンを用いた．実験結果を表 8 に示す．これらは 10 試行の平均値である．

天候予測スケジューリング手法は，欲張り法部分のみを用いた場合と比較し，（晴れ-雨）の場合に約 1.15 倍の期待値を持つようなスケジュール木を得ることが出来た．このことから，天候予測スケジューリング手法における局所探索法部分は，天気変化パターンがどのような場合にもうまく働くと考えられる．

3.5 まとめ

本章では，天気の変化に対応可能な観光スケジュール木算出問題に対して，欲張り法と局所探索法（n-swap 法）に基づいた近似アルゴリズムを提案した．この局所探索法は，部分木を単位とした目的地の置換を行うもので，同じ目的地を複

数巡回するスケジュールを含まないスケジュール群を生成する点に特色がある．先ず，天候予測スケジューリング手法を評価するため，20 の観光スポットを持つランダムに作成されたインスタンスを用いて実験を行った．その結果，本提案手法は，欲張り法を用いた場合と比べて平均 1.23 倍の期待値を持つスケジュール木を得ること，また交換回数 n は 2 回が適切であることがわかった．次に，天候予測スケジューリング手法により得られたスケジュール群を評価するため，シミュレーションにより従来手法との比較実験を行った．その結果，最も確率の高い天気と最も確率の低い天気を考慮して立てられたそれぞれのスケジュールに対し，天気が安定しない場合のインスタンスにおいて，7%から 10%上回る期待値を持つスケジュールを得ることができた．さらに，天候予測スケジューリング手法により得られた解の最適性を評価するため，6 つの観光スポットを持つような奈良県内観光インスタンスを用いて，全探索法と比較した．その結果，全探索で 18.3 時間かかるような問題に対して，4 秒以内で同一の解を発見することができた．

4. ユーザの体力変化に対応可能な観光スケジュール立案手法

4.1 概要

本章では，周辺のコンテキスト（観光場所，観光方式など）を考慮すると同時に，指定した観光方式の体力消費や休憩方式による体力回復を考慮し，旅行者の満足度最大を目指す観光スケジュール作成システムを提案する．同じ行動をとった場合の疲労度と回復度の個人差は大きく，各行動に対し，各旅行者それぞれの体力消費・回復が異なる．本手法では，人間が活動している時に消耗しているカロリー値を用いて，体力消耗を評価する．複数の観光地を巡回する最も満足度の高いスケジュールを算出するため，捕食法を利用したヒューリスティックアルゴリズムを提案する．このアルゴリズムは暫定解算出フェイズと近傍探索フェイズの二つのフェイズで構成される．まず暫定解算出フェイズでは，ランダムに生成した解を通常の局所探索により改善し，暫定解を得る．しかし暫定解算出フェイズは目的地ごとに欲張り値の高い観光方式を割り当てるので，局所解に陥る可能性がある．近傍探索フェイズでは，この暫定解に基づき，近傍範囲を設定して，その範囲内のみを探索し，暫定解より良い解が見つかる．本提案手法では，観光候補地 10 の場合，全探索で得られた解の 95.65%の満足度を有するスケジュールを 13 秒で得られることを確認した．

4.2 体力を考慮した最大満足度観光スケジューリング問題

本節では，体力を考慮した最大満足度の観光スケジュールを算出する問題を定義する．

4.2.1 問題概要

観光スケジュールを立案するために，まず，観光者と観光候補地の関係をモデル化する．

表 9 観光候補地に対する観光方式の一例

観光候補地	観光候補地 v_1			観光候補地 v_2		観光候補地 v_3	
観光方式	q_1	q_2	q_3	q_4	q_5	q_6	q_7
消耗体力値	-110	-200	-340	-500	-700	-120	-90
滞在時間 (hour)	1	1.5	2	1.5	2	1.5	1
満足度	40	60	75	60	80	90	70

表 9 に観光候補地および観光方式の例を挙げる．便宜上，消耗体力値は負の値として定義している．例えば，$q_1 = -110$，$q_3 = -340$ のように，同じ観光候補地でも観光方式が違うと，観光者の体力消耗の度合いが変化する．本問題では，観光者の体力が許す範囲で満足度が最大になる観光スケジュールを求めることを目的とする．図 7 はスケジュール例である．出発時間を 9 時，帰着時間を 15 時とする．観光者に対して一時的に算出された観光スケジュールを $v_1 \rightarrow v_2 \rightarrow v_3$ とする．観光候補地 v_1，v_2 に対し，観光方式 q_2，q_5 を選択し，その後，観光候補地 v_3 の観光方式として q_6（消耗体力値が -120）を選択すると（図 7(i)），体力値が負になるため，このようなスケジュールは実行不可能である．次に，図 7(ii) のように，観光候補地 v_3 の観光方式を q_7（消耗体力値が -90）に変更すると，体力内での観光は可能になるが，満足度は低くなる．一方，図 7(iii) のように，観光候補地 v_2 と v_3 の間に休憩を入れると，体力値が回復するため，体力内でかつ満足度の高い観光が可能になる．この場合，スケジュールの満足度は 230 となる．本問題の目的は，観光者の体力に関する制約条件を満たしつつ，スケジュールの満足度を最大化することである．休憩が必要な場合，観光後に同じ場所で休憩を行うよう設定する．表 10 は，体力考慮スケジューリング手法で求めた休憩を含むスケジュールの一例である．出発時間を 9 時，帰着時間を 15 時とする．体力値が 0 以下にならないように，適宜休憩が入っていることが分かる．

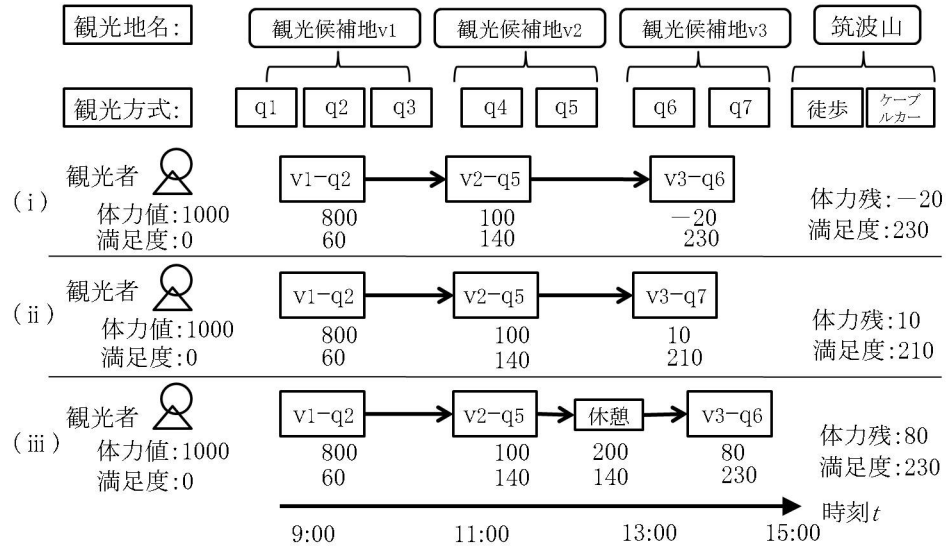

図 7 スケジュールの例

4.2.2 問題の定式化

本問題ではまず，2 つの集合 V, Q と関数 $dist(v_1, v_2)$ が与えられる．V は観光候補地（例：法隆寺，奈良公園など）の集合を，Q は観光方式の集合を表す．また，$Q_{v_i} \subseteq Q$ は観光候補地 $v_i \in V$ に対応した観光方式の部分集合を表す．$dist(v_1, v_2)$ は観光候補地 $v_1, v_2 \in V$ の間の距離を表す．スケジュール S を以下の式で定義する．

$$S = [d_0, ..., d_n]$$

ここで，d_j は j 番目に観光する観光候補地における観光方式と休憩時間を表しており，$d_j.v$，$d_j.q$，$d_j.r$ はそれぞれ観光地名，観光方式と観光後の休憩時間を示すこととする．

スケジュール S の満足度を与える関数 $Sat(S)$ を以下の式で定義する．ただし，$term1$ はスケジュールで得られる総満足度，$term2$ はホテル $(d_0.v)$ から出発し，各観光地 $(d_n.v)$ に訪問し，最後に，出発地点のホテル $(d_0.v)$ へ戻る移動距離に対

表 10 体力と満足度を考慮した観光スケジュールの例

到着時間	観光候補地	観光方式	満足度	体力値	出発時間	帰着時間
	Hotel	By Car	0.0	900.0	09:00	
09:04	Spot7	method1	101.4	644.0	10:34	
10:43	Spot5	method2	98.7	303.0	11:43	
11:43	Spot5	have rest	0.0	343.0	11:47	
12:00	Spot1	method1	93.1	10.0	13:00	
13:00	Spot1	have rest	0.0	210.0	13:20	
13:24	Spot10	method1	105.3	5.0	14:54	14:58
観光スケジュールの満足度：398.5						

するペナルティである．

$$Sat(S) = \underbrace{\sum_{j=0}^{n} d_j.q.sat}_{term1} - \underbrace{\alpha(\sum_{j=0}^{n-1} dist(d_j.v, d_{j+1}.v) + dist(d_n.v, d_0.v))}_{term2} \tag{10}$$

ここで $q.sat$ は観光方式 q の満足度，n は観光スケジュール内の観光候補地の数である．また，α は観光経路の短さ（無駄な経路の少なさ）をどの程度重視するかを示すパラメータである．つまり，移動距離 1km ごとに満足度は α 点ずつ減点される．

ユーザが目的地 d_j での観光終了直後に残っている体力値 $RemSt(d_j)$ は以下の式 (11) で与えられる．ただし，体力値はユーザの最大体力値 $MaxST$ を越えない．$term3$ は目的地 d_{j-1} での体力値と目的地 d_j での体力消耗値，回復する体力値の総和である．

$$RemSt(d_j) =$$

$$\begin{cases} \mathrm{Min}(\underbrace{RemSt(d_{j-1}) + d_j.q.conp + RstSt(d_{j-1})}_{term3}, MaxSt) & \text{if} \quad j > 0 \\ InitSt & \text{otherwise} \end{cases} \tag{11}$$

ここで，$q.conp$ は観光方式 q の体力消耗値，$MaxSt$ はユーザの最大体力，$InitSt$ は出発時の体力値である．また，目的地 d_j の観光後の休憩により，回復する体力値 $RstSt(d_{j-1})$ は以下の式 (12) で与えられる．$habitus$ は単位時間内に回復する体力値を表す．つまり，休憩時間 1 分ごとに体力値は $habitus$ 分回復する．

$$RstSt(d_{j-1}) = habitus \cdot d_{j-1}.r \tag{12}$$

スケジュールは以下の制約を満たさなければならない．

観光の間，ユーザの体力は負になってはならない．全目的地におけるユーザの体力に関する制約条件を式 (13) で示す．

$$\forall d_j.v \in [0, n-1], d_{j+1}.q.conp \leq RemSt(d_j) \tag{13}$$

スケジュール S における観光時間制約条件は式 (14) に示す．ただし，$term4$ はスケジュールで得られる総滞在時間，$term5$ はスケジュールで得られる総休憩時間，$term6$ は各観光候補地間の移動時間，$term7$ はホテルに戻るのにかかる移動時間である．

$$\underbrace{\sum_{j=0}^{n} d_j.q.stayt}_{term4} + \underbrace{\sum_{j=0}^{n} d_j.r}_{term5} + \underbrace{\sum_{j=0}^{n-1} \frac{dist(d_j.v, d_{j+1}.v)}{speed}}_{term6} + \underbrace{\frac{dist(d_n.v, d_0.v)}{speed}}_{term7}$$

$$\leq MaxTi \tag{14}$$

ここでは，$q.stayt$ は観光方式 q の滞在時間であり，$speed$ は二つのスポット間の移動スピードであり，$MaxTi$ は最大観光時間である．

以上を踏まえて本問題の目的関数を以下の式で定義する．

$$\begin{aligned} &\text{maximize} && Sat(S) \\ &\text{subject to} && restrictions(13) - (14) \end{aligned} \tag{15}$$

4.2.3 NP 困難性

複雑化された経路探索問題として一般化された巡回セールスマン問題 (GTSP)[31, 32] がある．これは 巡回セールスマン問題（TSP）において，全ての都市を幾つかのクラスタに分けたもので，この GTSP を TSP に変換できる手法 [33] も提案されている．またこの問題を効率良く解くため，幾つのヒューリスティックアルゴリズム [20, 21] が提案されている．

提案する問題は，一般化された巡回セールスマン問題 (GTSP) に出発／帰着点や時間制約，体力制約等を加えたものであり，GTSP を本問題に帰着することができる．GTSP は NP 困難であるから [33]，提案する問題も NP 困難である．

4.3 体力を考慮した観光スケジューリングアルゴリズム

4.2.3 項で定義した問題は NP 困難であるため，問題例の規模が大きい時には，実用時間内で最適解を求めることは難しい．そこで，捕食法を利用したヒューリスティックアルゴリズムを提案する．本節では，まず 4.3.1 項においてヒューリスティック手法である捕食法を紹介し，次に 4.3.2 項においてこれを利用した体力考慮スケジューリング手法について述べる．

4.3.1 捕食法

自然界の捕食獣は捕食行動が驚くほど類似している．捕食獣が捕食する時，複数の狩場をターゲットとし，順番に探索し，獲物が見つかるまで狩場をすばやく切り替える．獲物を発見すると，その狩場の最も獲物の多い場所に近づき，狩り易い獲物を選択し，これを追って捕獲する．獲物を逃した場合，次の狩場に変更する．動物の捕食行動を模した空間探索手法である捕食法は，Linhares が 1998 年に提案した [16]．図 8 に示すように，捕食法のアルゴリズムは次の 2 つの探索過程を含む:（1）広域探索－最初に全体の解空間内で，制約条件を満たす解のうち，暫定解を一つ求める．指定された繰り返し規定回数で暫定解が改善できない場合，アルゴリズムは終了する;（2）近傍探索－広域探索で見つかった暫定解に基づき近傍範囲を設定したうえで探索し，暫定解より良い解の有無を確認する．良い解

を発見した場合，近傍探索の範囲を新たに設定し，探索を繰り返す．設けた近傍探索範囲内で良い解が見つからなければ，近傍探索を放棄し，広域探索に戻る．これらの 2 つの探索は解を改善するため，交互に繰り返し適用される．近傍の大きさを調整することにより，広域探索と近傍探索のバランスを調整する点に特色がある．

Linhares は巡回セールスマン問題 (TSP)[16] および大規模集積回路 (VLSI) 設計問題 [18] に捕食法を適用した．また，Liu らは解の近傍を再定義することにより，捕食法を改良した [17]．体力考慮スケジューリング手法では，Linhares のオリジナルの捕食法 [16, 18] を改良し，4.2 節で定義した観光スケジューリング問題に適用する．

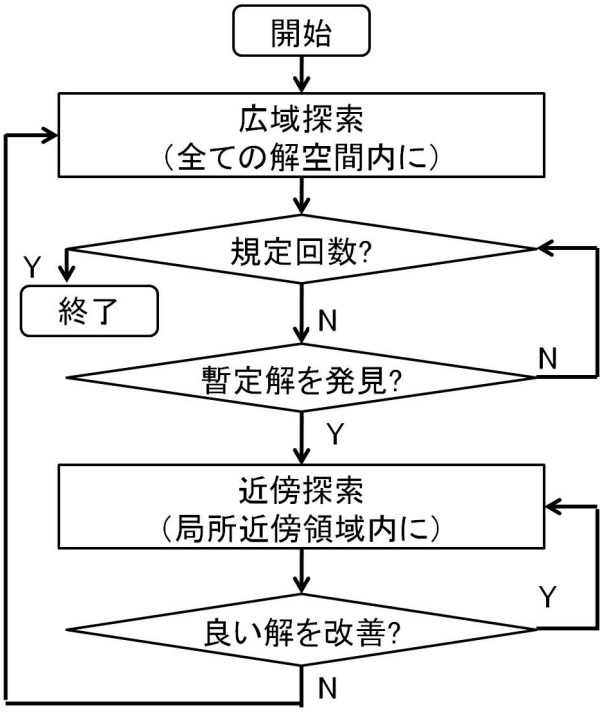

図 8 捕食アルゴリズム

Algorithm 1 *The proposed algorithm*

1: $S^{cm} \leftarrow Temporarysolutionsearchphase()$

2: $S \leftarrow Neighborhoodsearchphase(S^{cm})$

4.3.2 提案アルゴリズム

対象とする観光スケジューリング問題は，観光候補地，観光順，観光方式など，選択の対象が段階的に絞られるという特徴を持っている．本問題の解である観光スケジュールでは，各目的地 d を 4 項組 $d = \langle v, j, q, r \rangle$ で表す．ここで $v \in V$ は観光候補地, $j \in [0, n]$ はその観光候補地 v の巡回順序，$q \in Q$ は観光方式であり，また，r は観光後にとる休憩時間である．例えば，$\langle v_1, 2, q_2, 4 \rangle, \langle v_2, 1, q_1, 0 \rangle$ は，1 番目に観光候補地 v_2 を観光方式 q_1 を用いて観光してから，2 番目に観光候補地 v_1 を観光方式 q_2 を用いて観光し，4 分休憩するようなスケジュールを示す．観光順序と観光方式の決まっていない目的地は $\langle v_2, *, *, r \rangle$ のように記述する．

提案するアルゴリズムは（1）暫定解算出フェイズ,（2）近傍探索フェイズの二つのフェイズで構成される．まず暫定解算出フェイズ（捕食法の広域探索に相当）では，ランダムに生成した解を通常の局所探索により改善し，暫定解を得る．次に近傍探索フェイズ（捕食法の近傍探索に相当）では，ランダムに複数の部分空間を生成し，暫定解に近い部分空間から順に探索していく．暫定解よりも良い解が見つかった場合，その解を中心に部分空間を再生成し，改めて探索を行う．この過程は再帰的に適用され，有望な暫定解（狩場）の近傍をあたかも捕食獣が獲物を探すかのように探索することができる．全体的なアルゴリズムの擬似コードを Algorithm1 に示す．スケジュール S^{cm} と S はそれぞれ暫定解，最終解である．また，関数 $Temporarysolutionsearchphase()$ は暫定解算出フェイズ，関数 $Neighborhoodsearchphase()$ は近傍探索フェイズである．

以下，これらフェイズについて詳しく述べる．

4.3.2.1 暫定解算出フェイズ

巡回セールスマン問題では，すべての目的地を巡回する経路を求める．しかし

Algorithm 2 *Temporary solution search phase*

1: $n \leftarrow MaxTi/min_{q \in Q}\{q.stayt\}, \delta \leftarrow Maxloc$
2: $V' \leftarrow V, O \leftarrow \{1, ..., n\}, S^c \leftarrow \emptyset, S^{cm} \leftarrow \emptyset$
3: **while** $|S^c| < n$ **do**
4: select $v \in V'$ at random
5: select $j \in O$ at random
6: $q \leftarrow argmax_{q \in Q_v} PerCostSat(q)$
7: $S^c \leftarrow S^c \cup \{\langle v, j, q, * \rangle\}$
8: $V' \leftarrow V' - \{v\}, O \leftarrow O - \{j\}$
9: **end while**
10: $S^{cm} \leftarrow Modify(S^c)$
11: $S^{cm} \leftarrow LocalSearch(S^c, S^{cm}, \delta)$
12: **return** S^{cm}

本問題では，時間制約に基づいて巡回する目的地をいくつか選択しなければならない．従来の捕食法ではランダムに生成された初期解から探索を始めるが，体力考慮スケジューリング手法では探索効率を高めるため，まず欲張り法と局所探索法を利用して質の高い初期解（暫定解）を算出する．このフェイズの擬似コードを Algorithm2 に示す．

Algorithm2 では，まず，観光時間制約条件を満たすよう，最大目的地の数を決め，候補地からランダムに目的地を選択し，一つの観光スケジュールを生成する．次に，関数 $PerCostSat()$ を利用し，欲張り法で各目的地の観光方式を決める．それから，関数 $Modify()$ を適用し，適切に休憩を入れ，観光方式を含んでいる観光スケジュールの満足度を算出する．最後に，局所探索関数 $LocalSearch()$ を用いて解を改善する．

以下に疑似コードを用いて詳細に述べる．1 行目で巡回する目的地の数 n を計算する．ここでは，最大観光時間 $MaxTi$ および最小の観光時間 $q.stayt$ から，巡回しうる最大の目的地数を求める．定数 δ は局所探索の回数パラメータであり，予備実験により，$Maxloc = 5 \cdot |V|$ の値を用いている（後述）．2 行目では，観光候補地の集合 V'，観光順の集合 O，及びスケジュール比較解 S^c と暫定解 S^{cm} を

初期化する．3–9 行目では，巡回する観光候補地 $v \in V'$，観光順 $j \in O$ をランダムに決定し，目的地 $\langle v, j, q, * \rangle$ を構成する．6 行目で，欲張り値（式 (16) で後述）を計算し，最も高い欲張り値（次項参照）を持つ観光方式 q を，その観光候補地 v の観光方式とする．7–8 行目では，生成された目的地を S^c に追加する．S^c に属する目的地の数が n になるまで，上記のプロセスを繰り返す．また，8 行目では，任意の 2 つの目的地が同じ観光地や同じ観光順を含まないよう制約条件を更新する．例えば，観光候補地の集合 $|V| = 6$，巡回する目的地の数 $n = 4$ の場合，ランダムに生成する一つのスケジュールは $\{\langle v_0, 0, *, * \rangle \langle v_1, 2, q_2, * \rangle \langle v_2, 1, q_1, * \rangle \langle v_5, 4, q_1, * \rangle \langle v_3, 3, q_4, * \rangle\}$ となる．ここでは，出発／帰着ホテルは $\langle v_0, 0, *, * \rangle$ とする．10 行目では，目的地の休憩時間 r の追加や，制約時間を超える目的地の削除などの処理を手続き $Modify()$ を利用し，S^c を修正する．最後に，S^c を改善するため，局所探索法 $LocalSearch()$ を用いる．

以降，Algorithm2 内で用いる関数について述べる．

- $PerCostSat(q)$ **関数**

  関数 $PerCostSat(q)$ は，観光方式 q ごとに与えられる“ どれだけ効率的に満足度が得られるか（単位時間あたりの満足度）”の指標である．この値が大きいほど，観光地としての価値が高い．式 (16) でこの値の計算方式を示す．ただし，$q.conp/habitus$ は消耗体力に対する回復時間を表す．即ち，$PerCostSat(q)$ の値に対し，時間と消耗体力が影響を与え，滞在時間が長い，また消耗体力の大きい観光地は価値が低い．一方，ユーザの体力回復能力が“ 強ければ ”，消耗体力の影響が小さくなる．

$$PerCostSat(q) = q.sat/(q.stayt + abs(q.conp)/habitus) \tag{16}$$

  例えば，表 9 では，観光候補地 v_2 の観光方式 q_2 に対し，消耗体力値 $q_2.conp$ は -200，滞在時間 $q_2.stayt$ は $1.5hour$，満足度 $q_2.sat$ は 60 とする．もし観光者の休憩状態において毎分体力回復率 $habitus = 10$ と設定すると，単位時間あたりの満足度の結果は $PerCostSat(q_2) = 60/(1.5 * 60 + 200/10) = 0.55$ である．

- $Modify(S^c)$ **関数**

  関数 $Modify(S^c)$ は，スケジュール S^c が制約条件を満たすよう修正して返す関数である．体力の制約条件 (13) を満たすため，この関数は，スケジュール S^c の中に適切に休憩を入れる．j 番目に巡回する目的地 d_j で体力値が足りない場合，直前の目的地 d_{j-1} の休憩時間を以下の式で計算する．

  $$d_{j-1}.r = -RemSt(d_j)/habitus$$

  例えば，表 10 では，3 番目の観光候補地 $Spot9$ に巡回すると，体力値が足りなくなるため，直前の観光候補地 $Spot1$ において 4 分の休憩を行う．またスケジュールが帰着時間の制約 (14) を満たさない場合，満たすまで最後から一つずつ目的地を削除してゆく．この関数は，以上の処理によって修正されたスケジュールを返す.

- $LocalSearch(S^c, S^{cm}, \delta)$ **関数**

  関数 $LocalSearch(S^c, S^{cm}, \delta)$ は，スケジュール S^c 内の観光候補地をランダムに交換する局所探索を行う関数である．スケジュール内の目的地 d_i をランダムに選択し，また観光候補地 v' を V からランダムに選択する．選択された観光候補地 v' がスケジュール S^c 内の別の目的地 d_j に含まれていた時は，目的地 d_i と d_j の観光順を交換し，新たなスケジュール S'^c を得る．選択された観光候補地 v' が含まれていなかった時には，目的地 d_i の観光候補地を v' で上書きする．例えば，スケジュール $\{\langle v_0, 0, *, *\rangle \langle v_1, 2, q_2, *\rangle \langle v_2, 1, q_1, *\rangle \langle v_5, 4, q_1, *\rangle \langle v_3, 3, q_4, *\rangle\}$ 内の目的地 $\langle v_2, 1, q_1, *\rangle$ をランダムに選択し，また観光候補地 $\langle v_5, *, *, *\rangle$ を V からランダムに選択する．選択された観光候補地 $\langle v_5, *, *, *\rangle$ がスケジュール内に含まれているので，二つの目的地の観光順を交換し，スケジュール $\{\langle v_0, 0, *, *\rangle \langle v_1, 2, q_2, *\rangle \langle v_2, 4, q_1, *\rangle \langle v_5, 1, q_1, *\rangle \langle v_3, 3, q_4, *\rangle\}$ を得る．もし観光候補地 $\langle v_6, *, *, *\rangle$ が選択された場合，スケジュール $\{\langle v_0, 0, *, *\rangle \langle v_1, 2, q_2, *\rangle \langle v_6, 1, *, *\rangle \langle v_5, 4, q_1, *\rangle \langle v_3, 3, q_4, *\rangle\}$ を得る．また Algorithm2 の 6 行目と同様の方法を用い欲張り値が最大の観光方式を与えて，新たなスケジュール S'^c を得る．スケジュール S'^c に前述の関

数 $Modify(S'^c)$ を適用し，制約条件を満たすよう修正する．最後に変更前のスケジュール S^{cm} と変更されたスケジュール S'^c の満足度を比較し，改善されていれば，採用する．でなければ，破棄する．この改善の試みは δ 回数繰り返す．

4.3.2.2 近傍探索フェイズ

体力考慮スケジューリング手法では捕食法を利用し，観光スケジュールを観光候補地，観光順，観光方式の順番で決定する．ここではスケジュールの解空間を全体解空間，部分解空間，部分部分解空間，そして解の順で細分化してゆく．

- **全体解空間**：スケジュールの観光候補地，観光順，観光方式，休憩時間をすべて未定とした解空間であり，$\langle *, *, *, * \rangle^n$ で表す．
- **部分解空間**：巡回する観光候補地のみを決定したもの．その他（順番や観光方式）を未定とした解空間であり，$\langle v, *, *, * \rangle^n$ で表す．
- **部分部分解空間**：ある部分解空間でさらに，観光順番を決定したもの．観光方式は未だ未定であり，$\langle v, j, *, * \rangle^n$ で表す．
- **解**：ある部分部分解空間でさらに，観光方式と休憩時間を決定したもの．これは**観光スケジュール**，つまり一つの解であり，満足度が計算できる．

各解空間の探索方法をそれぞれ，Algorithm3，4，6 に示す．

Algorithm3 は，全体の解空間の探索を行うアルゴリズムである．暫定解算出フェイズで得られた解 S^{cm} を暫定解 S の初期値とする．観光候補地のみ含む複数の部分解空間をランダムに作成し，暫定解 S に属する空間と比較する．関数 $DistV()$ を適用し，比較結果として，複数の部分解空間をソートする．次に，各部分解空間内を順に探索する関数 $SubDomainSearch()$ を繰り返し呼び出して，得られた解が改善された場合には，暫定解 S を更新し，全部分解空間を再生成し，再度探索をする．生成された全ての部分解空間を探索しても暫定解 S が更新されなかった時，アルゴリズム全体を終了する．

以下に疑似コードを用いて詳細に述べる．3-6 行目で，l_{max} 個の部分解空間をランダムに作成している．ここでは $D_1, ..., D_{l_{max}}$ で表す．実験では，$\beta(= l_{max})$ の

Algorithm 3 *Neighborhood search phase*

1: $S \leftarrow S^{cm}, n \leftarrow MaxTi/min_{q \in Q}\{q.stayt\}, l_{max} \leftarrow \beta, D_k \leftarrow \emptyset, SearchDone \leftarrow false$
2: **while** $SearchDone = false$ **do**
3: **for all** k such that $1 \leq k \leq l_{max}$ **do**
4: generate D_k by selecting n $v \in V$ at random
5: $D_k.distv \leftarrow DistV(S, D_k)$
6: **end for**
7: generate $D'_1, ..., D'_{l_{max}}$ by sorting $D_1, ..., D_{l_{max}}$ in the increasing order of $D_k.distv$ $(1 \leq k \leq l_{max})$
8: $k \leftarrow 1, UpDated \leftarrow false$
9: **while** $k \leq l_{max}$ and $UpDated = false$ **do**
10: $S' \leftarrow SubDomainSearch(D'_k, n)$
11: **if** $Sat(S') > Sat(S)$ **then**
12: $S \leftarrow S', UpDated \leftarrow true$
13: **else**
14: $k \leftarrow k + 1$
15: **end if**
16: **end while**
17: if $k = l_{max} + 1$ then $SearchDone \leftarrow true$, and give up the neighborhood search
18: **end while**
19: **return** S

値は文献 [17] と [34] を参考に行った予備実験より $3n$ としている．部分解空間 D_k は巡回する n 個の目的地の観光候補地のみ含む．4 行目で，観光候補地のみから成る目的地をランダムに D_k に追加していく．具体的には，観光候補地集合 V からランダムに選択し，観光候補地のみからなる目的地を部分解空間に追加し，選択された観光候補地 v を観光候補地列から削除している．5 行目で，暫定解 S に近い部分解空間から探索するため，それぞれの部分解空間と，S に属する空間（観光候補地のみ）との間の距離を関数 $DistV()$ で計測する．暫定解 S に属する空間と部分解空間の距離は相似程度を表す．

例えば，暫定解算出フェイズで得られた暫定スケジュール S= $\{\langle v_0, 0, *, * \rangle \langle v_1, 2, q_2, 0 \rangle$

$\langle v_2, 1, q_1, 0\rangle$ $\langle v_5, 4, q_1, 5\rangle$ $\langle v_3, 3, q_4, 10\rangle\}$ とする．l_{max}= 3 × 4 個の部分解空間 $D_1 = \{\langle v_0, 0, *, *\rangle$ $\langle v_1, *, *, *\rangle$ $\langle v_2, *, *, *\rangle$ $\langle v_3, *, *, *\rangle$ $\langle v_5, *, *, *\rangle\}, ..., D_{12} = \{\langle v_0, 0, *, *\rangle$ $\langle v_3, *, *, *\rangle$ $\langle v_4, *, *, *\rangle$ $\langle v_5, *, *, *\rangle$ $\langle v_6, *, *, *\rangle\}$ をランダムに作成する．それぞれの部分解空間と，S に属する空間との間の距離は $D_1.distv = 0, ..., D_{12}.distv = 2$ である．関数 $DistV()$ の説明は後述する．7 行目で，計測された距離に応じ，距離の近いものから部分解空間をソートし，$D'_1, ..., D'_{l_{max}}$ とする．8–16 行目で，関数 $SubDomainSearch()$ を用いて，部分解空間内を順に探索していく．得られた更新解を S' とする．11–12 行目で，得られた解 S' が改善された場合には，暫定解 S を更新する．そして 3 行目に戻って全部分解空間を再生成し，更新された暫定解 S に近い部分空間から再度探索を開始する．最後の部分解空間まで探索しても改善されなければアルゴリズムは終了する．この時点での S が，全体アルゴリズムにより得られた解となる．

以降，Algorithm3 内で用いる関数について述べる．

- $SubDomainSearch$ **関数**

  関数 $SubDomainSearch(D, n)$ は，部分解空間 D 内の探索を行う関数である．この関数の擬似コードを Algorithm4 に示す．ここで n は，スケジュールに含まれる目的地の個数である．Algorithm4 では，まず，Algorithm3 で生成した部分解空間内に，関数 $TemporarySolution()$ を適用し，この部分解空間の暫定解 S を生成する．次に，観光候補地の観光順番を決定する複数の部分部分解空間をランダムに作成し，暫定解 S に属する観光順空間と比較する．関数 $SDistV()$ を適用し，Algorithm3 と同様な探索方法で複数の部分部分解空間をソートする．それから，各部分部分解空間内を順に探索する関数 $RandomSelectionInSubSubDomain()$ を繰り返し呼び出す．ここで得られた解が改善された場合には，暫定解 S を更新し，全部分部分解空間を再生成し，再度探索をする．生成された全ての部分部分解空間を探索しても暫定解 S が更新されなかった時，Algorithm4 を終了する．

  以下に疑似コードを用いて詳細に述べる．まず観光スケジュール S，観光順 O, 及びフラグ $SearchDone$ を初期化する．2 行目では，関数 $TemporarySolution()$ を用いて，暫定解 S を生成する．関数 $TemporarySolution()$ の擬似コード

Algorithm 4 *SubDomainSearch Function*

1: $S \leftarrow \emptyset, O \leftarrow \{1, ..., n\}, m_{max} \leftarrow \gamma,\ SD_k \leftarrow \emptyset, SearchDone \leftarrow false$
2: $S \leftarrow TemporarySolution(D, n)$
3: **while** $SearchDone = false$ **do**
4: **for all** k such that $1 \leq k \leq m_{max}$ **do**
5: generate SD_k by selecting $v \in \{v' | \langle v', *, *, * \rangle \in D\}$ and $j \in O$ at random
6: $SD_k.distv \leftarrow SDistV(S, SD_k)$
7: **end for**
8: generate $SD'_1, ..., SD'_{m_{max}}$ by sorting $SD_1, ..., SD_{m_{max}}$ in the increasing order of $SD_k.distv$ $(1 \leq k \leq m_{max})$
9: $k \leftarrow 1, UpDated \leftarrow false$
10: **while** $k \leq m_{max}$ and $UpDated = false$ **do**
11: $S' \leftarrow RandomSelectionInSubSubDomain(SD'_k, n)$
12: **if** $Sat(S') > Sat(S)$ **then**
13: $S \leftarrow S', UpDated \leftarrow true$
14: **else**
15: $k \leftarrow k + 1$
16: **end if**
17: **end while**
18: if $k = m_{max} + 1$ then $SearchDone \leftarrow true$, and give up the subdomain search
19: **end while**
20: **return** S

を Algorithm5 に示す. 関数 $TemporarySolution()$ は暫定解算出フェイズとほぼ同様の処理を行うので，Algorithm2 の説明を参照する．ここでは，予備実験により，$Maxloc = 3 \cdot |V|$ の値を用いている．

4–8 行目では，部分解空間 D の中に，複数の部分部分解空間 $SD_1, ..., SD_{m_{max}}$ を生成する．つまり，同じ観光候補地を巡回する中で，巡回順序を m_{max} 種類だけ設定する．γ は部分部分解空間の生成個数を示すパラメータである．実験では，文献 [17] と [34] および予備実験により，γ は n とする．また，6 行目で，S に属する観光順空間と SD_k の間の距離を関数 $SDistV()$ で計測する．例えば，関数 $TemporarySolution()$ を用いて，暫定解 S= $\{\langle v_0, 0, *, *\rangle$

Algorithm 5 *TemporarySolution Function*

1: $\delta \leftarrow Maxloc, S^{cm} \leftarrow \emptyset$
2: **while** $|S| < n$ **do**
3: select $v \in \{v' | \langle v', *, *, * \rangle \in D\}$ at random
4: select $j \in O$ at random
5: $q \leftarrow argmax_{q \in Q_v} PerCostSat(q)$
6: $S \leftarrow S \cup \{\langle v, j, q, * \rangle\}$
7: $V' \leftarrow V' - \{v\}, O \leftarrow O - \{j\}$
8: **end while**
9: $S^{cm} \leftarrow Modify(S)$
10: $S^{cm} \leftarrow LocalSearch(S, S^{cm}, \delta)$
11: **return** S^{cm}

$\langle v_1, 1, q_2, 0 \rangle \langle v_2, 2, q_1, 0 \rangle \langle v_3, 3, q_3, 10 \rangle \langle v_4, 4, q_4, 5 \rangle\}$ を生成する．$m_{max} = 4$ 個の部分部分解空間 $SD_1 = \{\langle v_0, 0, *, * \rangle \langle v_1, 2, *, * \rangle \langle v_2, 1, *, * \rangle \langle v_3, 4, *, * \rangle \langle v_4, 3, *, * \rangle\}, ..., SD_4 = \{\langle v_0, 0, *, * \rangle \langle v_1, 2, *, * \rangle \langle v_2, 1, *, * \rangle \langle v_3, 3, *, * \rangle \langle v_4, 4, *, * \rangle\}$ をランダムに作成する．それぞれの部分部分解空間と，S に属する観光順空間との間の距離は $SD_1.distv = 3, ..., SD_4.distv = 2$ である．関数 $SDistV()$ の説明は後述する．8 行目で，計測された距離に応じ，距離の近いものから部分部分解空間をソートする．次に，9–17 行目では，これら部分部分解空間内 SD'_k を順に，関数 RandomSelectionInSubSubDomain() を呼び出して探索する．得られた解 S' が改善された場合には，暫定解 S を更新する．そして 4 行目に戻って全部分部分解空間を再生成し，更新された暫定解 S に近い部分部分解空間から再度探索を開始する．最後の部分部分解空間まで探索しても改善されなければ，フラグ $SearchDone$ は *true* に設定し，S が Algorithm4 により得られた解となる．

- $DistV$ と $SDistV$ 関数

体力考慮スケジューリング手法では，部分解空間や部分部分解空間を解空

間の近さによりソートする．ここでは，この近さを与える二つの関数を定義する．

1. 関数 $DistV$ は以下の式で定義される．

$$DistV = n - v_{same} \tag{17}$$

ここでは，v_{same} は二つのスケジュールの内の同じ観光候補地の数であり，n は経路の目的地の数である．例えば，二つのスケジュール $S1 = \{d_0.v_0, d_1.v_1, d_2.v_2, d_3.v_3, d_4.v_4, d_5.v_5\}$，$S2 = \{d_0.v_0, d_1.v_3, d_2.v_4, d_3.v_5, d_4.v_6, d_5.v_7\}$ の間の近さを計算する．同じ観光候補地は v_0，v_3，v_4，v_5 なので，$v_{same} = 4$，$DistV(S1, S2) = 6 - 4 = 2$ となる．

2. 関数 $SDistV$ は以下の式で定義される．

$$SDistV = n - j_{same} \tag{18}$$

ここでは，j_{same} は二つのスケジュールの内に存在する同じ“辺”（向きを考えない）つまり目的地間の隣接の数である．また，n はスケジュールの“辺”の数である．以下の二つのスケジュール $S1'$ と $S2'$ では，目的地間の“辺”は“−”で表示する．例えば，二つのスケジュール $S1' = \{d_0.v_0 - d_1.v_1 - d_2.v_2 - d_3.v_3 - d_4.v_4 - d_5.v_5 - d_0.v_0\}$，$S2' = \{d_0.v_0 - d_1.v_2 - d_2.v_1 - d_3.v_3 - d_4.v_5 - d_5.v_4 - d_0.v_0\}$ の間の近さを計算する．同じ辺は $d_1.v_1 - d_2.v_2$, $d_4.v_4 - d_5.v_5$（ここでは，前後が逆でも同一の“辺”であるとみなす）なので，$j_{same} = 2$，$SDistV(S1', S2') = 6 - 2 = 4$ となる．

- $RandomSelectionInSubSubDomain$ **関数**

関数 $RandomSelectionInSubSubDomain(SD, n)$ は，Algorithm4 で生成した部分部分解空間 SD 内において観光方式をランダムに決定したスケジ

Algorithm 6 *RandomSelectionInSubSubDomain Function*

1: $S \leftarrow \emptyset$
2: **while** $|S| < n$ **do**
3: select $d \in \{d' | \langle v', j', *, * \rangle \in SD\}$ at random
4: select $q \in Q_{v'}$ at random
5: $S \leftarrow S \cup \{\langle v, j, q, * \rangle\}$
6: **end while**
7: $S \leftarrow Modify(S)$
8: **return** S

ュールを返す関数である．この関数の擬似コードを Algorithm6 に示す. 最後に，観光候補地，観光順番と観光方式を決定したスケジュールに対し，関数 $Modify()$ を適用し，適切に休憩を入れる観光スケジュールの満足度を算出する．

提案アルゴリズムは発見的アルゴリズムであり，前回のループで発見されたものよりも改善された解が発見され続ける限りアルゴリズムは終了しない．そのため，最悪の場合の計算量は $O(n!)$ になる．

4.4 評価実験

体力考慮スケジューリング手法の性能を確認するために，Java 言語でシミュレータを作成し，コンピュータシミュレーション実験を行った．シミュレーションを実行した PC は CPU が Celeron 2GHz，メインメモリが 2GB，OS が Windows XP である．体力考慮スケジューリング手法の性能を確認するに当たり，解の最適性及び演算時間それぞれの観点において，本提案手法と複数ある参考手法（4.4.1 項）とを比較し，体力考慮スケジューリング手法の性質を考察する（4.4.2 項）.

1. 解の最適性：スケジュールの満足度を指標とし，異なるサイズの候補地集合において，全探索を含む複数の参考手法と比較し，各種手法で出力した

スケジュールの満足度をどれほど最適値に近づけるかを考察した.

2. 演算時間：各種手法が解を求めるまでの演算時間を指標とし，良い解を実用時間内に求められるかについて考察した.

観光者の体力設定として，最大体力値 $MaxSt$ を 900 とし, 休憩状態において毎分体力回復率 $habitus = 10$ と設定した．実験では，13500m × 9000m のフィールドに観光候補地をランダムに含んだインスタンスを計 10 個を用いた：

- インスタンス [小]：マップ id1〜5，含まれる観光候補地の数は 10.
- インスタンス [大]：マップ id6〜10，含まれる観光候補地の数 20.

各候補地の重要度は 1-100 の値をランダムで設定した．実験を簡単化するため，各候補地の観光方式を 2 種類とした．また各観光方式の滞在時間は，1 時間，1.5 時間及び 2 時間とした．表 11 はこれらの設定に基づく候補地例の一つである．候補地の観光方式は計 6 通りあり，観光方式ごとの消耗体力値は統計情報 [30] に基づいて設定した．また，移動方式を自動車と想定し，すべての実験において，目的地間の移動速度を 30km/h とした．なお，移動距離に対する制御パラメータ α を 3 とし，移動距離が 1km になるごとに満足度は 3 点減点される.

表 11 観光候補地の例

観光方式	滞在時間	満足度	体力消費
山を登る a	1h	27	410
山を登る b	1.5h	72	615
山を登る c	2h	85	819
ジョギングと歩行の組合せ a	1h	48	341
ジョギングと歩行の組合せ b	1.5h	79	512
ジョギングと歩行の組合せ c	2h	75	682

4.4.1 参考手法

体力考慮スケジューリング手法の性能を評価するため，複数の参考手法と比較実験を行った.

- **欲張り法**：出発地から移動した後，次の目的地・観光方式の満足度を最大化することだけを考えて選択する手法．観光者の体力がなくなると，休憩を入れて，帰着時刻まで目的地を巡回する．この手法は次の目的地の満足度だけに注目するため，移動距離の総和を最短化しようとしない.
- **GA 法**：複数日にわたる観光のためのスケジュール作成手法 [15] を利用した手法．元の手法は体力を考慮していなかったが，体力考慮スケジューリング手法と同様に体力を考慮して休憩を挿入するよう修正している.
- **PS 法**：暫定解算出フェイズを用いず，ランダムに生成された解から近傍探索フェイズのみで改善する手法．従来の捕食法 (Predatory Search, PS) に相当する.
- **TSS 法**：体力考慮スケジューリング手法の暫定解算出フェイズのみを用いた方法．TSS とは $Temporary\ Solution\ Search$ の略記である．TSS 法を実装する際，妥当な欲張り関数が必要である．複数の欲張り関数候補を予備実験（詳細実験結果を割愛）で求めた結果，3.2.1.1 で挙げた関数 $PerCostSat()$ を採用している.
- **全探索法**：すべての組み合わせを全探索により求め，最大の満足度のスケジュールを得る方法.

4.4.2 シミュレーション実験

4.4.2.1 解の満足度

算出スケジュールの満足度を評価するために，5 種類の参考手法との比較実験を行った．また，大小両方の観光地集合（インスタンス）を実験に用いることで，体力考慮スケジューリング手法のスケーラビリティを確認した．それぞれの実験

結果は 30 試行の平均となっている．また，GA 法と PS 法に対し，探索時間を制限し，演算時間が同じ条件のもとで比較を行った．

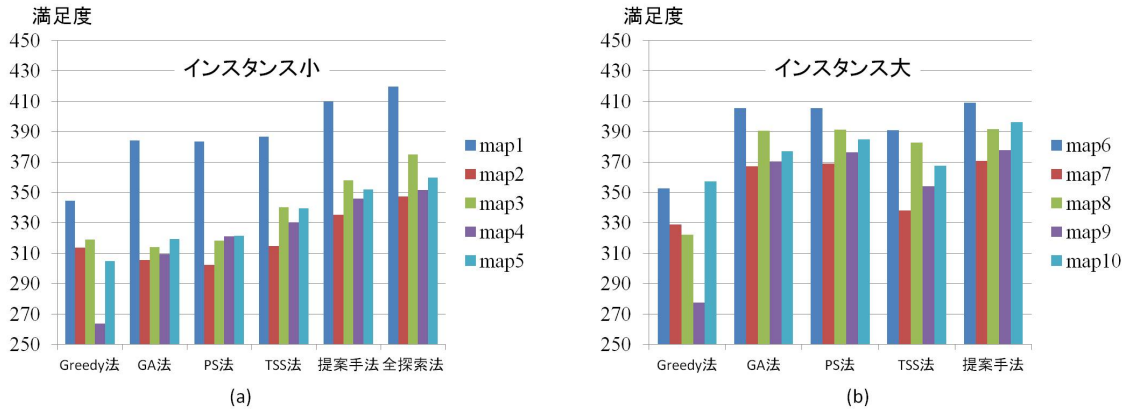

図 9 満足度の比較結果

インスタンス [小] の結果を図 9(a) と表 12 に示す．全探索法では，最適解を求めるのに 109.75 分かかったのに対し，体力考慮スケジューリング手法は約 13 秒で最適解の 95.65% 以上の満足度を持つ解を求めることができた．GA 法と PS 法では，体力考慮スケジューリング手法よりも長い演算時間を設定したにも関わらず，得られた満足度は本提案手法より低い．一方，Greedy 法と TSS 法は短い時間で演算を終えるが，解の質は低く，局所解に陥っていると考えられる．欲張り法と比較したところ，体力考慮スケジューリング手法は，1.36 倍の満足度を持つ解を得ることが出来た．

インスタンス [大] の結果は図 9(b) と表 12 に示されている．全探索法で最適解を求めることは実用時間内で不可能なため，その結果を掲示していない．体力考慮スケジューリング手法と他の手法を比較した結果，ほぼ同様の満足度の解を得るまでに，体力考慮スケジューリング手法では 93.75 秒，GA 法と PS 法で 160 秒程度かかった．体力考慮スケジューリング手法は解の質を維持しつつ，GA 法と PS 法より明らかに計算量が少ないことが分かる．これと比べ，TSS 法と欲張り法はインスタンス [小] の場合と同様な傾向が見られ，短い演算時間で演算を求められるが，解の質が低い．

表 12 平均計算時間の比較結果

平均計算時間	欲張り法	GA 法	PS 法	TSS 法	提案手法	全探索法
インスタンス小	0.6ms	19.43s	20.16s	0.52s	13.68s	109.75min
インスタンス大	10.5ms	164.21s	167.45s	6.28s	93.75s	—

4.4.2.2 計算時間の評価

体力考慮スケジューリング手法の計算時間を評価するため，全探索の計算時間と比較した．実験には，ランダムに生成されたインスタンス 10 個を用いた．10 個のインスタンスに含まれる観光地候補数は 6-24 個（6,8,10,12,14,16,18,20,22,24）である．また，各候補地の観光方式の満足度もランダムに生成した．全探索は探索候補数が増えるにつれ，計算時間が爆発的に増加するため，観光候補地数（6,8,10,12）の場合についてのみ計算時間を計測した．実験結果を表 13 に示す．候補地数 12 の場合，計算時間が 7.08 時間かかることがわかった．図 10 に，体力考慮スケジューリング手法の計算時間（30 試行の平均）を示す．候補地数 18 以下の場合，1 分以内で解を得ることができていることが分かる．体力考慮スケジューリング手法の計算時間は候補地数が 20 まで，緩やかに伸びている．各試行における計算時間のばらつきも大きくない．しかし，体力考慮スケジューリング手法は候補地数が 22 の時，計算時間自体はさほど長くなっていないが，各試行の計算時間のばらつきが大きくなり，候補地数が 24 の時，計算時間が 300 秒近くまで膨れ上がった一方，計算時間のばらつきもさらに顕著になった．これは，候補地数が多いほど，経路探索時の運の影響が大きくなっているからである．体力考慮スケジューリング手法は最悪の場合の計算量が $O(n!)$ になると考えられ，より多くのインスタンスでこれを実証する予定であるが，さらにインスタンスを拡大することは実用時間の観点からみると困難と思われる．

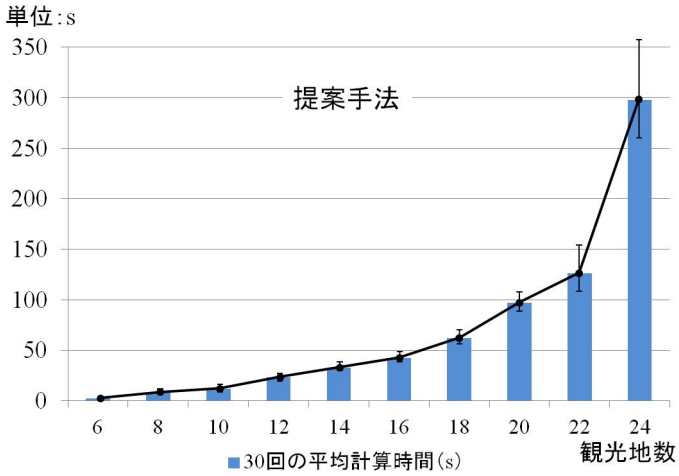

図 10 体力考慮スケジューリング手法の計算時間評価

表 13 全探索法の計算時間評価

インスタンス	候補地数 (6)	候補地数 (8)	候補地数 (10)	候補地数 (12)
全探索法	1.37min	18.44min	106.2min	424.93min

4.5 まとめ

本章では，観光者の体力と観光時間を制約とし，合計の満足度を最大化するような観光スケジュールの立案手法を提案した．体力考慮スケジューリング手法は，TSP の有効解法の一つである捕食法を利用している点に特色がある．解の探索効率を評価するため，体力考慮スケジューリング手法と全探索で求めた解とを比較した．その結果，最適解の 95.65%以上の満足度を持つ解を平均 12.99 秒で得ることができた．また，欲張り法と比較したところ，20 個の目的地数を持つインスタンスにおいて 1.36 倍の満足度を持つ解を得ることが出来た．

今後の課題として，移動中の体力を考慮することが挙げられる．厳しい天候の元では，移動経路が屋外か屋内か，またバス等を利用できるかが体力消耗に大き

な影響を与えることが考えられ，これらの利用スケジュールをうまく立案することは重要である考えられる．また，目的地の観光自体が体力の回復につながる場合が考えられる．例えばレストランで食事をしたり，温泉で休憩するなど，目的地での休憩も考慮したスケジューリング手法を提案する予定である．

本章では，観光方式ごとの消耗体力値を，統計情報 [30] に基づいて設定した．具体的には，観光者の体力消費として，人間が活動している時に消耗しているカロリー値を用いた．例えば，観光者の初期体力値としては，一般人がある一つの活動を最長時間で続ける時，消耗したカロリー値を設定した．しかし実際には，同じ観光方式をとった場合の消耗体力値と回復度の個人差は大きいと考えられる．また，観光方式の数が多いとき，観光者が旅行の度にそれらの値を全て入力することは繁雑である．人の感覚に基づいたインスタンスを設定するために，アンケートを用いての各観光方式の疲労度設定が有用であると考えられる．

5. 結論

本論文では，観光におけるコンテキストの大きな要素として，天気と体力を考慮した二つの観光スケジュール作成支援について取り扱い，二つの手法を提案した．以下，得られた成果および今後の課題と考察をまとめる．

3 章の天気の変化に対応可能な観光スケジュール群立案手法では，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し，スケジュール木における満足度の期待値の総和最大化を目的とする問題を取り扱った．この問題を解くため，新しいヒューリスティック手法 n-swap 法を提案した．n-swap 法は Lin-Kernighan 法で用いられる 2-opt や 3-opt 法と似ているが，スケジュール木の部分木内において同じ観光地を割り当てられたノードを調べて交換するもので，同じ目的地を複数回にわたって巡回しないシーケンス（一つのスケジュール）の組合せから成るスケジュール群を生成できる点に特色がある．この観光案内システムを用いて，観光者はあらかじめスケジュール群を立てておいて，実際の天気に応じて利用する部分を選ぶことができる．天候予測スケジューリング手法を評価するため，20 の観光スポットを持つランダムに作成されたインスタンスを用いて実験を行った．その結果，本提案手法は，欲張りを用いた場合と比べて平均 1.23 倍の期待値を持つスケジュール木を得ること，また交換回数 n は 2 回が適切であることがわかった．また，天候予測スケジューリング手法により得られた解の最適性を評価するため，6 つの観光スポットを持つような奈良県内観光インスタンスを用いて，全探索法と比較した．その結果，全探索で 18.3 時間かかるような問題に対して，4 秒以内で同一の解を発見することができた．

4 章のユーザの体力変化に対応可能な観光スケジュール立案手法では，周辺のコンテキスト（観光場所，観光方式など）を考慮すると同時に，指定した観光方式の体力消費や休憩方式による体力回復を考慮し，旅行者の満足度最大化を目的とする問題を取り扱った．この問題を解くため，動物の捕食行動を模した探索アルゴリズム捕食法 (PS 法) を利用した．捕食法のアルゴリズムは近傍の大きさを調整することにより，広域探索と近傍探索のバランスを調整する点に特色がある．この観光案内システムを用いて，ユーザが好きな観光方式を選択できると同時に，体力に合わせて最適となる観光プランを算出することができる．体力考慮

スケジューリング手法により得られた解の最適性を評価するため，10 観光候補地を有するインスタンスに対し提案手法と全探索で求めた解を比較した．その結果，最適解の 95.65%以上の満足度を持つ解を平均 12.99 秒で得ることができた．また，欲張り法と比較したところ，20 か所の観光候補地を持つインスタンスにおいて 1.36 倍の満足度を持つ解を得ることが出来た．

本研究では，天気と体力を考慮した二つの観光スケジューリング手法を提案した．提案手法では，全ての候補地から適切な目的地を選択している．一方，実際の観光スケジュールでは“絶対に巡回したい目的地”が存在する場合がある．その目的地に極めて高い満足度を与えることにより，それを含むスケジュールを生成させることができるものの，探索効率が悪くなるおそれがある．この案内機能を追加するため，必ず訪れる目的地を予め選択し，その後，他の候補地に対して探索を行う．さらに，提案アルゴリズムを用いて，スケジュール中各要素の位置を変換することにより，探索効率を向上させることが可能と考えられる．また，本研究は個人観光を対象としたものであった．しかし，グループ観光においては，ユーザ体力の制約条件が特に顕著となる．全員の体力を同時に考慮せねばならない上，分離合流を伴うそれぞれのユーザの経路を求めなければならないため，問題の複雑度が増すと考えている．体力と観光方式を考慮したグループ観光スケジュール作成問題が今後発展していくと予想される．現在 iPhone などのスマートフォンが注目を集め急速に普及してきている．本研究により開発するシステムが算出するスケジュールに従い，GPS 機能を備えた携帯端末を介し，旅行者にナビゲーション機能を広く提供する為には，スマートフォンアプリケーションの開発・配布が必要である．収集した利用状況から提案手法の有効性やさらなる改善点を解析することも含めて今後の課題としたい．旅行者にとって過度な負担もなく満足度の高い観光ができると共に，観光中の天気変化や交通渋滞などの突発的な出来事に対応した観光スケジュールの変更もできるようになり，より観光しやすくなることが期待される．現実世界において，各観光地は健康条件，営業時間や定休日が異なる場合あり，これらは目的地の集合に反映され，提案手法に直接影響を与えない．しかし，スケジュール内目的地の順番を変更する際，問題が発生可能性がある．現時点では，提案手法は全観光地に対して一様に取り扱っており，個別の

観光地に対する特殊な扱いに対応していない．

本研究の具体的な応用先として，電気自動車の充電と走行問題がある．現在，自動車のエンジンは，地球温暖化の防止と石油依存からの脱却に向けて，電動化の動きが急速に進展している．しかし電気自動車は，満充電から走行可能な距離が短いという制約が存在する．長距離のドライブのためには，バッテリ残に対する走行可能距離と急速充電スタンドの所在地から，どこで充電すれば良いかを示すスケジュールが必要になる．提案手法はこの目的に利用できる可能性が高い．また，ダイエットナビやゲームなどに応用することも可能である．コンテキストアウェアナビケーションシステム以外の組合せ最適化問題に関する研究だけではなく，一般的な生活応用に対応することも，重要な課題となり，有用性が高いと思われる．

謝辞

本研究を進めるにあたり，伊藤　実　教授には，御多忙にも関わらず主指導教員となって頂きました．謹んで感謝の意を表します．また，入学前におきましても，留学を認めていただき，日本で研究するという道を与えていただきました．深く感謝致します．さらに，入学後，学術研究の以外にも，学生生活の様々な部分において御援助を頂きました．心から御礼申し上げます．

安本　慶一　教授には，本研究を行うにあたり，熱心な御指導と多大な御助力を頂きました．特に難問に直面した際に，貴重なご意見を頂き，問題を乗り越えることができました．また，研究のみに関わらず，多くの対外発表の機会を与えて頂き，生活面でもお世話になりました．ここに深く感謝し，厚く御礼申し上げます．

関　浩之　教授には御多忙にも関わらず，論文審査委員を引き受けて頂き，本論文の作成についても貴重なご意見を頂きました．謹んで感謝の意を表します．

柴田　直樹　准教授には，研究を進める中で，御助言と御指導を頂きました．謹んで感謝の意を表します．

広島市立大学　情報科学研究科　村田　佳洋　准教授には，遠方にも関わらず，ビデオ会議やメールで多くの御助言，御指導を頂きました．本研究の中で，アルゴリズムの考案，プログラムの作成などにおいて，熱心な御指導を頂きました．また，多くの対外発表の際に，温かい教示を頂きました．学術論文の執筆時にも多大な御支援を頂きました．深い感謝と共に，厚く御礼申し上げます．

孫　為華 助教には，本研究を進めるにあたり，多くの御助言を頂きました．ここに深く感謝致します．

また，尾川　恵理，金岡 恵　事務補佐員には，研究費管理などの事務手続き等をお世話して頂きました．有難うございました．

最後になりましたが，伊藤研究室でお世話になりました全ての皆様，NAIST で学生生活を共に過ごせた全ての皆様に感謝の念と御礼を申し上げます．末筆ながら，皆様のご多幸をお祈り致します．

参考文献

[1] 国土交通省観光庁, “統計情報, ”
http://www.mlit.go.jp

[2] Baus, J., Krüger, A. and Wahlster,W., “A Resource Adaptive Mobile Navigation System,” Proc. of 2002 Int’l. Conf. on Intelligent User Interfaces 2002 (*IUI-02*), pp. 15–22 (2002).

[3] Butz, A., Baus, J., Krüger, A. and Lohse, M., “A Hybrid Indoor Navigation System,” Proc. of 2001 Int’l. Conf. on Intelligent User Interfaces 2001 (*IUI2001*), pp. 25–33 (2001).

[4] Cheverst, K., Davies, N., Mitchell, K., Friday, A. and Efstratiou, C., “Developing a Context-aware Electronic Tourist Guide: Some Issues and Experiences,” Proc. of 2000 ACM Special Interest Group on Computer-Human Interaction (*SIGCHI-00*) pp. 17–24 (2000).

[5] Rehrl, K., Leitinger, S., Bruntsch,S. and Mentz, H. J., “Assisting Orientation and Guidance for Multimodal Travelers in Situations of Modal Change,” Proc. of 2005 IEEE Int’l. Conf. on Intelligent Transportation Systems (*ITSC-05*) pp. 407–412 (2005).

[6] Maruyama, A., Shibata, N., Murata, Y., Yasumoto, K. and Ito, M., “P-Tour: A Personal Navigation System for Tourism,” Proc. of World Congress on ITS, pp. 18–21 (2004).

[7] Wu, B., Murata, Y., Shibata, N., Yasumoto, K. and Ito, M., “A Method for Composing Tour Schedules Adaptive to Weather Change,” Proc. of the 2009 IEEE Intelligent Vehicles Symposium (*IV’09*), pp. 1407–1412 (2009).

[8] Cheverst, K., Davies, N., Mitchell, K. and Friday, A., “The Design of an Object Model for a Context-Sensitive Tourist Guide,” *Computers Graphics Journal*, Vol. 23, No. 6, pp. 883–891 (1999).

[9] 福田義文，新吉高，田中克明，“複数アプリケーション間における入力フォーカスを管理する Java 応用車載端末プラットフォームの開発,” マルチメディア，分散，協調とモバイル (DICOMO2005) シンポジウム論文集，Vol. 2005, No. 6, pp. 45–48 (2005).

[10] Korf, R., “Real-Time Heuristic Search,” *Artificial Intelligence*, Vol. 42, No. 2–3, pp. 189–211 (1990).

[11] Kawamura, H., Kurumatani, K. and Ohuchi, A.,“Modeling of Theme Park Problem with Multiagent for Mass User Support,” *Multiagent for Mass User Support*, Vol. 3012, No. 004, pp. 48–69 (2003).

[12] Miyauchi, H., Shiga, K., Omoto, M., Hamanaka,T. and Kano, K., “Style of Providing Information on Tourist Resort,” World Congress on Intelligent Transport Systems, pp. 3146–3153 (2004).

[13] Sakagawa, M., Takeuchi, A., Nishio, I. and Inoue, T., “Pedestrian Navigaion System in Nagoya,” World Congress on Intelligent Transport Systems, pp. 3373-3380 (2004).

[14] Helsgaun, K., “An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic,” *European Journal of Operational Research*, Vol. 126, No. 1, pp. 106–130 (2000).

[15] 木下 隆正, 永田 宗伸, 村田 佳洋, 柴田 直樹, 安本 慶一, 伊藤 実：複数日にわたる観光のための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.47, No.12, pp.3179-3187 (2006).

[16] Linhares, A., “State-space Search Strategies Gleaned from Animal Behavior: a Traveling Salesman Experiment,” Biological Cybernetics, Vol. 78, No. 3, pp. 167-174 (1998).

[17] Liu, C. and Wang, D., “Predatory Search Algorithm with Restriction of Solution Distance,” Biological Cybernetics, Vol. 92, No. 5, pp. 293-302 (2005).

[18] Linhares, A., “Synthesizing a Predatory Search Strategy for VLSI Layouts,” IEEE Trans. on Evolutionary Computation, Vol. 3, Iss. 2, pp. 147-152 (1999).

[19] Ahuja, R. K., Ergun, O., Orlin, J. B. and Punnen, A. P., “A Survey of Very Large-scale Neighborhood Search Techniques,” *Discrete Applied Mathematics 123*, pp. 75–102 (2002).

[20] Iordache, S., “Consultant-Guided Search - A New Metaheuristic for Combinatorial Optimization Problems,” Proc. of 2010 Genetic and Evolutionary Computation (GECCO’10), ACM Press (2010).

[21] Petrica, C. P. and Iordache, S., “A Hybrid Heuristic Approach for Solving The Generalized Traveling Salesman Problem,” Proc. of 2011 Genetic and Evolutionary Computation (GECCO’11), pp. 481-488 (2011).

[22] Suthikarnnarunai, N., “A Sweep Algorithm for the Mix Fleet Vehicle Routing Problem,” Proc. of the Int’l MultiConf. of Engineers and Computer Scientists(*IMECS 2008*), Vol. II, pp. 1914–1919 (2008).

[23] Ismail, Z. and Irhamah, “Solving the Vehicle Routing Problem with Stochastic Demands via Hybrid Genetic Algorithm-Tabu Search,” *Journal of Mathematics and Statistics*, Vol. 4, Issue 3, pp. 161–167 (2008).

[24] Gajpal, Y. and Abad, P., “An Ant Colony System (ACS) for Vehicle Routing Problem with Simultaneous Delivery and Pickup,” *Computers and Operations Research*, Vol. 36, Issue 12, pp. 3215–3223 (2009).

[25] 橋本英樹，今堀慎治，柳浦睦憲，茨木俊秀，“移動時間コスト関数を考慮した時間枠つき配送計画問題に対する局所探索法,” 数理解析研究所講究録，Vol. 1349, pp. 94–112, (2004).

[26] Giaglis G. M., Minis I., Tatarakis A. and Zeimpekis V., “Minimizing Logistics Risk Through Real-time Vehicle Routing and Mobile Technologies: Research

to Date and Future Trends," *International Journal of Physical Distribution and Logistics Management*, Vol. 34, Issue 9, pp. 749–764 (2004).

[27] Guo, Z. G. and Mak, K. L., "A Heuristic Algorithm for The Stochastic Vehicle Routing Probrems with Soft Time Windows," Proc. of the Congress on Evolutionary Computation(*CEC 2004*), Vol. 2, No. 5, pp. 1449–1456 (2004).

[28] Yahoo Japan, "Yahoo! the weather information, "
http://weather.yahoo.co.jp/weather

[29] Tang, F., You, I., Guo, M. and Guo, S., "Context-aware Workflow Management for Intelligent Navigation Applications in Pervasive Environments," Intelligent Automation and Soft Computing, Vol. 16, No. 4, pp. 605-619 (2010).

[30] Club.Panasonic, "運動・生活活動の消費カロリー, "
http://club.panasonic.jp/diet/exercise/mets/index.html

[31] Tasgetiren M. F., Suganthan P. N. and Pan Q. K., "A Discrete Particle Swarm Optimization Algorithm for the Generalized Traveling Salesman Problem," Proc. of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO'07), pp. 158-167 (2007).

[32] Hu B. and Raidl G. R., "Effective Neighborhood Structures for the Generalized Traveling Salesman Problem," Proc. of Evolutionary Computation in Combinatorial Optimization (EvoCOP 2008), LNCS, Vol. 4972, pp. 36-47 (2008).

[33] Behzad A. and Modarres M., "A New Efficient Transformaiton of The Generalized Traveling Salesman Problem into Traveling Salesman Problem," Proc. of the 22rd International Conference on Software Engineering (ICSE 2002), pp. 43-46 (2002).

[34] Kenneth, D. B., Andrew, B. K. and Sudhakar, M., "A New Adaptive Multi-start Technique for Combinatorial Global Optimizations," Operations Research Letters, Vol. 16, Iss. 2, pp. 101-113 (1994).

業績リスト

論文誌

1. 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,” 情報処理学会論文誌 数理モデル化と応用, Vol.3，No.1，pp.87-97 (January 2010).
（3 章）

2. 武　兵，孫　為華，村田佳洋，安本慶一，伊藤　実：“ ユーザの体力変化に対応可能な観光スケジュールの立案手法，” 情報処理学会論文誌 (採録決定).
（4 章）

国際会議 (査読有り)

1. Bing Wu ○, Yoshihiro Murata, Naoki Shibata, Keiichi Yasumoto, and Minoru Ito:“ A Method for Composing Tour Schedules Adaptive to Weather Change,” Proc. of 2009 IEEE Intelligent Vehicles Symposium (IV’09), pp. 1407-1412 (June 2009).
（3 章）

2. Bing Wu ○, Yoshihiro Murata, Naoki Shibata, Keiichi Yasumoto, and Minoru Ito:“ A Stamina-Aware Sightseeing Tour Scheduling Method,” Proc. of 2011 Genetic and Evolutionary Computation (GECCO’11), pp. 81-82 (July 2011).
（4 章）

国内学会 (査読有り)

1. 武　兵 ○, 村田 佳洋, 柴田 直樹, 安本 慶一, 伊藤　実 :“ 天気の変化に対応可能な観光スケジュール作成手法,” マルチメディア，分散，協調とモバイル (DICOMO2008) シンポジウム, pp. 1651 - 1660 (July 2008).
（3 章）

2. 武　兵○, 村田 佳洋, 柴田 直樹, 安本 慶一, 伊藤 実:“ 天候を考慮した観光スケジュール群の局所探索を用いた立案手法,” マルチメディア，分散，協調とモバイル (DICOMO2009) シンポジウム, pp. 473 - 479 (July 2009).
（3 章）

3. 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,” 情報処理学会 数理モデル化と問題解決 (MPS) 第 75 回研究会，北海道大学（札幌） (Sep 2009).
（3 章）

国内学会 (査読無し)

1. 武　兵○，孫為華, 村田佳洋，安本慶一，伊藤　実 : “ 満足度と体力を考慮した観光スケジューリング法の提案,” 情報処理学会 アルゴリズム (AL) 第 133 回研究会，愛媛大学（愛媛） (Jan 2011).
（4 章）