RESEARCH

Optimizing Group Utility in Itinerary Planning: A Strategic and Crowd-Aware Approach

Junhua Liu[1], Aldy Gunawan[2], Kristin L. Wood[1,3] and Kwan Hui Lim[1*]

*Correspondence: kwanhui_lim@sutd.edu.sg
[1]Singapore University of Technology and Design, Singapore
Full list of author information is available at the end of the article

Abstract

Itinerary recommendation is a complex sequence prediction problem with numerous practical applications. The task becomes significantly more challenging when optimizing multiple factors simultaneously, such as user queuing times, crowd levels, attraction popularity, walking durations, and operating hours. These factors, combined with the dynamic and unpredictable nature of visitor flow, introduce substantial complexities, particularly when accounting for collective user behavior. Existing solutions often adopt a single-user perspective, overlooking critical challenges arising from natural crowd dynamics. For example, the Selfish Routing problem illustrates how individual decision-making can lead to suboptimal outcomes for the group as a whole. To address these challenges, we propose the Strategic and Crowd-Aware Itinerary Recommendation (SCAIR) algorithm, which integrates real-world crowd behavior into route planning to optimize group utility. SCAIR models itinerary recommendation as a Markov Decision Process (MDP) and incorporates a novel State Encoding mechanism that facilitates real-time, efficient itinerary planning and resource allocation in linear time. By prioritizing group outcomes over individual preferences, SCAIR explicitly mitigates the adverse effects of selfish routing. We conduct extensive evaluations of SCAIR using a large-scale, real-world theme park dataset, benchmarking it against several competitive and realistic baselines. Our results demonstrate that SCAIR consistently outperforms these baselines, effectively addressing the limitations of selfish routing and significantly enhancing overall group utility across four major theme parks.

Keywords: Itinerary recommendation; Crowd-aware Algorithms; State Encoding; Utility Optimization; Markov Decision Process; Sequence Modelling

1 Introduction

The field of itinerary recommendation has experienced significant growth in recent years, driven by its broad applicability across various domains, particularly in tourism where efficient tour planning is crucial. Itinerary recommendation is inherently a complex sequence prediction problem, presenting substantial challenges in real-world scenarios. Solving such problems using exact optimization approaches is often infeasible due to the interdependence between data points, as each point influences others within the sequence. Consequently, the combinatorial search space expands exponentially as the dataset size increases, making exact solutions computationally prohibitive. To address this, most research relies on heuristic and function approximation methods, which are commonly employed to derive efficient and practical solvers for such problems.

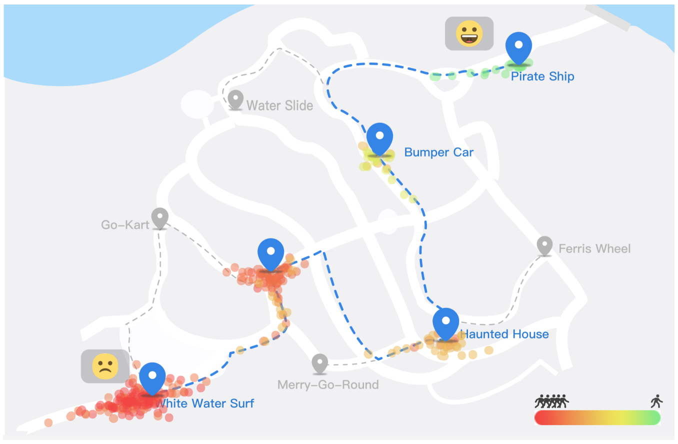

Figure 1 Current itinerary recommendation approaches predominantly employ data-driven methods that focus on single-user perspectives. However, this often leads to the Selfish Routing problem in real-world scenarios, where allowing individuals to act solely based on their preferences results in sub-optimal social welfare for the collective group. As illustrated, the recommended path is performed sub-optimally, where overcrowding occurs at POIs closer to the start of the route, while other POIs remain underutilized, as depicted by the greyed-out areas.

1.1 Motivation and Real-World Challenges

Recommending and planning effective itineraries in real-world environments is particularly complex and challenging due to the involvement of multiple Points of Interest (POIs), each with varying levels of popularity and crowd density. For instance, in a theme park, a visitor's route might include attractions such as roller coasters, water rides, and other attractions or live events. Itinerary recommendation can be framed as a utility optimization task, where the objective is to maximize the number and popularity of visited facilities [1], while simultaneously minimizing queuing times and travel times between facilities. Facilities in a theme park exhibit diverse properties such as popularity, duration of stay, location, and dynamic queuing times. Furthermore, visitors are often constrained by a fixed time budget, limiting the number of attractions they can visit in a single trip. Although numerous algorithms have been developed [1, 2, 3, 4, 5] to recommend itineraries, they primarily focus on individual travelers. However, in real-world scenarios, an itinerary is also influenced by the actions of other visitors, such as increasing queuing times at popular facilities due to crowding.

To develop an effective route planning strategy, data on past visit histories of attractions can be obtained from publicly available sources, such as Flickr, Wikipedia, or Google Reviews. This data can then be used to analyze the popularity, expected queuing times, and distances between various facilities. Personalizing the strategy further involves incorporating user preferences to create a customized itinerary [6]. However, in real-world dynamic environments, visitors often struggle to identify an optimal path due to the lack of real-time information about other visitors' movements. Similarly, static recommendation algorithms based solely on historical data fail to optimize for social welfare, as they do not account for the current state of the

[1]We use the terms "POIs", "attractions," and "facilities" interchangeably in this paper.

environment at the time of recommendation. For example, visitors typically adopt one of two intuitive strategies: (1) minimizing travel distance by selecting the nearest facility or (2) maximizing the popularity of attractions by visiting the next most popular location within a certain radius. Our research demonstrates that neither of these strategies effectively maximizes a visitor's utility in dynamic systems, where the actions of other visitors significantly impact the overall state of the system.

1.2 Existing Approaches and the Selfish Routing Problem

Many existing works focus on constructing a single optimal path for individual travelers, based on historical data. While this approach may work well for a single traveler, it becomes suboptimal when all travelers receive the same recommendation. Consider a recommender system that suggests an itinerary comprising the most popular POIs with the historically shortest queuing times. In a real-world scenario with multiple travelers, all agents will follow the same recommended itinerary, initially benefiting from the short historical queuing times. However, as more travelers arrive and follow this path, the expected queuing times will increase, as shown in Figure 1. In other words, the later an agent [2] arrives, the longer their expected queuing time will be. This situation leads to a failure in optimizing collective utility or social welfare for all agents. From the perspective of an individual traveler, it is difficult to gain knowledge of the system's state, i.e., the number of people visiting the park and their respective paths. As a result, allowing an agent to independently find an optimal strategy that maximizes their utility is unrealistic without considering the actions of other agents within the system.

While numerous algorithms have been developed for itinerary recommendation [2, 7, 4, 5], they predominantly focus on single-user optimization, assuming each traveler acts in isolation. These models often use historical data to recommend a sequence of popular or nearby attractions, optimizing for criteria such as distance [2], popularity [4], or visit recency [5]. However, in real-world scenarios, such as theme parks, multiple travelers concurrently move around, resulting in emergent crowd dynamics that are not captured by these static, individual-centric models. For example, algorithms like those in [2, 1] assume that queuing time is constant or estimated from past data, without accounting for how concurrent visitors following similar recommendations can inflate these times. Even group itinerary models, such as [8, 9], often emphasize preference aggregation within small predefined groups, and do not model system-wide interaction effects across multiple groups or individuals. As a result, these approaches may yield good individual outcomes but suboptimal collective outcomes, as they neglect the feedback loops between individual decisions and overall system state, a phenomenon well-known in game theory as the Selfish Routing problem [10].

The Selfish Routing problem refers to the phenomenon where individuals make route choices based solely on personal benefit, such as minimizing their own travel or queuing time, without regard for how their choices affect others. While such behavior may seem rational at the individual level, it often leads to system-wide inefficiencies, including overcrowding at popular attractions and under-utilization of other attractions. This problem has been widely studied in algorithmic game

[2] We use the terms *travelers*, *visitors*, and *agents* interchangeably in this paper.

theory, where it is shown that decentralized decision-making can produce outcomes far from the global optimum, a concept formalized by the Price of Anarchy [10]. In the context of itinerary recommendation, when all visitors are guided by the same personalized strategy (e.g., to the attraction with historically low queuing times), it creates congestion at those attractions, degrading the experience for everyone. Our work addresses this challenge by shifting from a purely individual optimization framework to a crowd-aware, system-level approach, in which recommendations are dynamically adjusted to account for real-time visitor flow and collective welfare.

1.3 Our Proposed Approach

To address this issue, we propose the Strategic and Crowd-Aware Itinerary Recommendation (SCAIR) algorithm, a recommender system that monitors internal information for all recommended routes and utilizes this information to provide dynamic routing recommendations to arriving agents. By adopting a game-theoretic approach, we develop a crowd-aware itinerary recommendation algorithm that mitigates the Selfish Routing problem [10], where allowing agents to act independently leads to suboptimal social welfare. Specifically, we model the itinerary recommendation problem as a strategic game [11], where the system (e.g., a theme park) establishes a set of allocation rules to assign routes to each player, rather than allowing agents full autonomy in choosing their paths. Our experiments show that this approach significantly improves group utility across agents, enhancing overall social welfare.

We conduct our experiments using a publicly available theme park dataset from [6], which comprises over 655k geo-tagged photos from Flickr. This dataset is notable for being the first to include queuing time distributions for attractions across various Disney theme parks in the United States. Preliminary experiments conducted on two smaller theme parks demonstrate promising results, with our approach significantly outperforming benchmark algorithms. However, we also identified a limitation of the proposed algorithm: the computation time required for the pathfinding process grows in factorial time with the size of the theme parks, presenting scalability challenges.

To overcome this scalability issue, we introduce a State Encoding mechanism that significantly reduces the computational complexity of the pathfinding algorithm. By leveraging the transition matrix of a Markov Decision Process, we efficiently track crowd distribution at each time step. This innovation allows us to reduce the algorithm's complexity from factorial to polynomial, making it computationally feasible for larger datasets. Consequently, we are able to conduct experiments on two larger theme parks, further demonstrating the scalability of our approach.

2 Main Contributions

The main contributions of this paper are summarised as follows [3]:

[3] This paper is an extended version of [12], with an addition of more than 50% new material. These additions include: (1) an updated literature review with more recent works and two additional domains, namely Vehicle Network and Natural Language Processing; (2) a more detailed description of the SCAIRv1's core algorithms with

- We introduce and formulate the crowd-aware itinerary recommendation problem as a social welfare optimization problem that accounts for the actions of multiple travelers, contrasting with existing approaches that focus solely on the single-traveler perspective (Section 4).
- To address this crowd-aware itinerary recommendation problem, we propose the SCAIR algorithm, which recommends itineraries that optimize group utility for multiple agents, effectively considering the collective impact of individual actions (Section 5).
- We propose a general state encoding mechanism that enables real-time itinerary recommendations in large environments, improving scalability and computational efficiency (Section 6).
- We conduct an algorithmic complexity analysis of the state encoding mechanism, demonstrating linear-time complexity for both the update procedure and the pathfinding algorithm (Section 6.6).
- We update and add new information to the original dataset to better facilitate our experiments and enhance the usability of the dataset. This dataset is made publicly available at `https://github.com/junhua/SCAIR` to facilitate further research on itinerary recommendation (Section 7.1).
- Using a theme park dataset, we compare our SCAIR algorithm against various competitive and realistic baselines and show how SCAIR outperforms these baselines with a large reduction in queuing times and improvement in utility (Sections 7 and 8).

The remainder of this paper is organized as follows: Section 3 reviews related works and highlights how our research extends and differs from these earlier contributions. Section 4 formulates the itinerary recommendation problem, incorporating crowd and queuing time awareness. Sections 5 and 6 present two versions of the strategic itinerary recommendation algorithms and introduce a system encoding mechanism. Section 6.6 provides a complexity analysis of the proposed algorithms. Sections 7 and 8 describe the experimental setup and discuss the results. Finally, Section 9 summarizes the paper's contributions and outlines potential future research directions.

3 Related Work

The field of itinerary recommendation has garnered significant attention, leading to a diverse array of proposed solutions spanning multiple disciplines. This review synthesizes key contributions from domains such as Operations Research, Vehicle Routing, Natural Language Processing, and Information Retrieval, emphasizing their relevance to itinerary recommendation and related tourism challenges. We perform a literature review of these studies to highlight the methodologies employed, including optimization techniques, heuristic algorithms, and data-driven models.

pseudo codes; (3) proposal of a new pathfinding algorithm and its state encoding mechanism with pseudo codes and extensive explanations; (4) an analysis of the algorithmic complexity; (5) additional experiments conducted on on two theme parks with 25 and 27 POIs, respectively, which are 45% to 108% larger than the two theme parks from previous work; (6) a more thorough discussion of experimental results, findings, and future research directions.

Furthermore, we identify the limitations inherent in existing approaches, particularly their challenges in optimizing group utility and accommodating dynamic real-world constraints. This discussion establishes a foundational context for introducing the novel methodology proposed in this paper, which aims to address these gaps and advance the state-of-the-art in itinerary recommendation systems.

3.1 Recent advancements of Itinerary and Tourism-related Recommendation

Numerous methodologies have been proposed to address the itinerary recommendation problem, leveraging variants of the Orienteering Problem to incorporate diverse constraints and objectives. For instance, Chen et al. introduced a multi-task learning framework that optimizes Points of Interest (POIs) selection by deriving consensus among group members, thereby enhancing group satisfaction [8]. Halder et al. proposed a Monte Carlo Tree Search-based reinforcement learning approach to prioritize POIs based on factors such as long visit durations, short queuing times, high popularity, and elevated visitor interest [13]. Sarkar et al. addressed group itinerary recommendation by modeling it as a non-Markovian process, incorporating travelers' historical data when available to tailor recommendations [9]. Zhang et al. employed heuristic approximations to solve a variant of the problem that accounts for POI opening hours and integrates uncertainties arising from diverse travel modes [14, 1]. Alternative strategies include leveraging the Ant Colony System to optimize itinerary planning [15], as well as adaptations that consider crowd levels to mitigate congestion at POIs [16]. Moreover, integer programming has been utilized to optimize itineraries by aligning with user interests, factoring in the duration of tourist visits at POIs as a measure of relevance [2].

Although recent advancements in itinerary recommendation have introduced important techniques to capture user preferences, group consensus, and contextual constraints, there remain certain limitations. For example, most existing works assume a static or single-agent setting, which fails to account for crowd interactions and real-time dynamics. These limitations restrict their effectiveness in environments like theme parks, where simultaneous decision-making among many agents can significantly alter the system state. Our work addresses this gap by introducing a dynamic, crowd-aware model that explicitly accounts for inter-agent dependencies and real-time queuing feedback.

3.2 Operations Research

The itinerary recommendation problem has often been modeled within the Operations Research domain as a variant of the Orienteering Problem. This problem represents routing challenges on a connected graph, where the objective is to determine a path through nodes that maximizes overall profit without exceeding predefined budget constraints. Typical budget constraints include travel time or distance between attractions in an itinerary [17, 18, 19, 20]. Solutions to the Orienteering Problem frequently focus on optimizing social welfare by maximizing global rewards, such as the popularity of Points of Interest (POIs). However, these approaches often neglect the trade-offs between a facility's visit duration and its popularity, a factor that can significantly influence the overall profit. Given the extensive history of research on the Orienteering Problem, several valuable survey papers comprehensively document its advancements and challenges over different periods [21, 22, 23].

These surveys provide critical insights into the evolution of methodologies, emerging trends, and persistent challenges, offering a holistic view of this problem space.

Approaching itinerary recommendation from the Operations Research perspective offers valuable problem formulations, particularly via orienteering variants that model path planning with budgets and constraints. While these approaches inform key constraints in our model, such as time and distance, they generally focus on optimizing static objectives and do not address dynamic system behavior caused by concurrent and multiple agents. In contrast, our approach integrates real-time crowd feedback into the itinerary generation process, enabling responsive and adaptive recommendations that optimize both individual and social welfare.

3.3 Vehicle Networking

In the domain of Vehicle Networking, numerous studies have addressed routing and scheduling challenges by formulating and solving variants of the Vehicle Routing Problem (VRP). These solutions leverage advanced algorithms, including the Genetic Algorithm [24], Tabu Search [25], Variable Neighborhood Search (VNS) [26], Simulated Annealing [27], and Branch and Cut [28]. Recent contributions in this area exemplify the application and enhancement of these algorithms. For instance, Han et al. introduced an improved Adaptive Genetic Algorithm that outperforms conventional genetic algorithms in efficiency and solution quality [24]. Li et al. utilized Tabu Search for gateway assignment in airport scheduling, demonstrating its effectiveness in optimizing resource allocation [25]. Similarly, Cai et al. proposed a collaborative Variable Neighborhood Search (VNS) framework tailored for multi-objective distributed scheduling tasks, showcasing improved performance in balancing conflicting objectives [26]. Other researchers have also studied different aspects of the VRP and related problem, such as in terms of considering a stochastic travelling time [29], charging prediction for electric buses [30]

Routing techniques from vehicle networking provide inspiration for scalable path optimization under multiple constraints. However, most VRP solutions are grounded in centralized planning for fleet management rather than decentralized, agent-driven itinerary selection. Moreover, crowd effects and mutual interference among agents are rarely modeled explicitly. Our study adapts the strengths of these algorithms, such as heuristic efficiency, while embedding them within a strategic framework that accounts for competitive agent behavior and social impact.

3.4 Natural Language Processing

In the Natural Language Processing (NLP) domain, we have seen a rapid advancement in sequence models in recent years following the introduction of attention mechanism and transformer models. In 2017, Vaswani *et al.* from Google introduced Transformer [31], a new category of deep learning models solely attention-based and without convolution and recurrent mechanisms. Later, Google proposed the Bidirectional Encoder Representations from Transformers (BERT) model [32], which drastically improved state-of-the-art performance for multiple challenging Natural Language Processing (NLP) tasks. Since then, multiple transformer-based models have been introduced, such as GPT [33], XLNet [34], T5 [35] and PaLM [36], among others. Transformer-based models were also deployed to solve domain-specific tasks, such as medical text inference [37], semi-structured data embedding [38], crisis signal detection [39], and occupational title embedding [40], and demonstrated remarkable performance. Despite promising results across different tasks in natural language processing, understanding, and inference, limited works examine the performance of transformer-based models in the itinerary recommendation space.

Advances in NLP have contributed powerful tools for sequence modeling and decision prediction, which can be useful for forecasting user trajectories or encoding complex state transitions. However, most transformer-based models in this space focus on individual preference inference and lack explicit mechanisms for modeling collective impact or crowd dynamics. Our work bridges this gap by combining MDP-based decision processes with dynamic, real-time state encoding, drawing inspiration from sequence modeling while extending it to a multi-agent, strategic context.

3.5 Information Retrieval

In the Information Retrieval community, item recommendation has long been as a prominent research area, with natural extensions into the domain of recommending Points of Interest (POIs). One widely explored approach involves leveraging algebraic techniques such as matrix factorization and tensor models to develop tourism recommender systems, effectively capturing latent user preferences and POI characteristics [41, 42]. Another prevalent methodology is top-k POI recommendation, which employs collaborative filtering augmented with additional mechanisms to generate a ranked list of the most relevant locations. These enhancements include incorporating contextual information, user preferences, and spatial-temporal factors to refine recommendations [43, 44, 45, 46, 47, 48].

Information retrieval approaches have enriched itinerary planning through collaborative filtering and tensor-based recommendation. However, they largely operate in a passive prediction mode and lack an interactive or strategic optimization layer. These systems tend to recommend high-utility POIs without accounting for the emergent effects of many users pursuing similar recommendations. In contrast, our algorithm dynamically adjusts recommendations to anticipate crowd effects, ensuring balanced usage of facilities and improving global system utility.

3.6 Overall Limitations

A notable limitation of earlier works is their reliance on recommendation algorithms constructed from an individual-centric perspective. While some recent studies have begun exploring the effects of group or crowd behavior [16, 49, 50, 51, 52], these approaches generally model the system as a static environment, where dynamic factors such as queuing times are derived solely from historical data. This static modeling framework has an inherent drawback: self-interested agents optimize their own objectives without accounting for the collective impact on social welfare. For example, in the context of a theme park, if all visitors are directed to follow the same recommended path, queuing times at popular attractions will surge, leading to a breakdown in the effectiveness and optimality of the recommendation system. This phenomenon aligns with the Selfish Routing problem discussed extensively by Roughgarden [10], where individual agents acting solely in their self-interest result in sub-optimal outcomes for the broader system. Addressing such challenges requires rethinking recommendation strategies to balance individual preferences with collective welfare, ensuring robust performance in dynamic, real-world environments.

The Selfish Routing problem has been extensively studied in the fields of Game Theory and Mechanism Design [10, 53, 54]. A key measure of inefficiency in such systems is the Price of Anarchy (PoA), defined as the ratio between the worst-case Nash equilibrium and the optimal collective payoff in a game-theoretic environment [53, 54]. This metric quantifies the disparity between self-interested decision-making and the global optimum. A well-known example of such inefficiency is Braess's Paradox [55], which illustrates how adding a new link to a transportation network can paradoxically increase the overall travel time for all users, as self-interested agents disrupt system-wide efficiency [56]. To address such issues, system operators can intervene by designing policies or economic incentives that guide agents toward more socially optimal outcomes. These interventions often involve leveraging game-theoretic principles to align individual actions with system-wide objectives. In this paper, we introduce a game-theoretic, dynamic itinerary recommendation algorithm as an example of such a strategy. By incorporating dynamic feedback and carefully designed mechanisms, our approach aims to balance individual preferences with collective welfare, mitigating inefficiencies and improving the overall effectiveness of the recommendation system.

3.7 Proposed Method

To address these limitations, we propose the Strategic and Crowd-Aware Itinerary Recommendation (SCAIR) algorithm, which aims to enhance welfare optimization by mitigating the inefficiencies caused by decentralized decision-making [57]. SCAIR dynamically models itinerary planning scenarios, such as theme park navigation, by considering all visitor itineraries and leveraging knowledge of other visitors' predicted paths. The algorithm recommends the next destination for each visitor while accounting for the expected queuing times at all facilities, which are dynamically updated based on the anticipated number of visitors at each location during a given time slot. A key innovation of SCAIR is the introduction of a State Encoding mechanism, which captures real-time crowd distribution data within a transition matrix. This mechanism enhances the algorithm's path-finding capabilities by providing a dynamic and granular view of crowd behavior. To validate its effectiveness, we evaluate SCAIR against three benchmark algorithms using simulations informed by real-world data. The results are analyzed and discussed to demonstrate the performance and scalability of our approach in optimizing both individual experiences and collective welfare.

4 Crowd-aware Itinerary Recommendation Problem

In this section, we first give an overview of our general approach, followed by formulating our crowd-aware itinerary recommendation problem, before showing the NP-hardness of this proposed problem.

4.1 General Approach

In this work, we approach the itinerary recommendation problem from a global perspective, formulating it as a strategic game. In this formulation, the system acts as a central planner, designing and distributing optimal paths for each agent upon arrival, based on the current state of the system and the paths of existing agents. In the context of a theme park, this central entity can be envisioned as the park operator, responsible for recommending customized itineraries to visitors to optimize their experience while balancing system-wide efficiency. To achieve this, we propose the Strategic and Crowd-Aware Itinerary Recommendation (SCAIR) algorithm, which dynamically generates route recommendations by accounting for the movements and interactions of all active agents in the system. By integrating real-time data and strategic modeling, SCAIR ensures that recommendations are adaptive to crowd distributions, enhancing both individual visitor satisfaction and overall operational efficiency.

The crowd-aware itinerary recommendation problem aims to maximize the collective utility of all agents in the system, effectively formulating a social welfare optimization problem. This problem is known to be NP-hard [58], presenting significant computational challenges. Moreover, simulating or solving this problem is empirically demanding, as it requires consideration of the entire history of existing visitors. This results in computational complexity that scales factorially with both the number of agents in the system and the number of facilities in a given path, making efficient solutions difficult to achieve.

To address these challenges, we propose a simplified model that frames the recommendation problem as a finite Markov chain. This approach leverages the computational efficiency of Markov models, which are known to be in NC [59] and decidable in poly-logarithmic time [60]. The simplified model assumes that each decision incorporates information about the immediately preceding decision, effectively encoding a snapshot of the entire decision history within the current state. This abstraction significantly reduces computational complexity while retaining sufficient information to guide accurate recommendations. In the following section, we detail the formulation of this problem.

4.2 Problem Formulation

We formulate the crowd-aware itinerary recommendation problem as a finite Markov chain, incorporating practical constraints to ensure alignment with real-world scenarios. Specifically, we impose the following constraints: (1) a fixed starting point, typically located near the entrance; (2) a time budget for the itinerary, reflecting the limited duration available for visitors to tour; and (3) a maximum allowable distance between two consecutive stations, addressing the dissatisfaction associated with long walking distances between facilities. These constraints enhance the model's applicability to real-life contexts, improving its relevance and usability for dynamic itinerary planning.

We model the theme park comprising numerous tourist attractions as a fully connected graph $G(F, C)$, where $F = \{f_1, ..., f_n\}$ is the collection of n facilities in the system, and $C = [c_{ij}]$ is the set of connections from f_i to f_j. Each connection c_x is associated with the properties of distance $Dist(c_{ij})$ and travel time $Trav(c_{ij})$ in minutes. Each facility f_x is associated with a set of properties including coordinates $(lat_x, long_x)$, duration of visit $Dur(f_x)$ in minutes, capacity $Cap(f_x)$ and popularity $Pop(f_x)$.

We formulate the agents' visits as m states $S = \{s_1, ..., s_m\}$, where each state s_x is associated with a feasible path $p_x = [f_1^{(x)}, ..., f_{n_x}^{(x)}]$ with n facilities $[f_1^{(x)}, ..., f^{(x_n)}]$. The total time TT_x of path p_x is defined as:

$$TT_x = \sum_{i=1}^{n_x} Dur(f_i^{(x)}) + \sum_{i=1}^{n_x-1} Trav(c_{i,i+1}) \tag{1}$$

We model the utility of agents as a function of the popularity of each facility visit, normalized by the expected waiting time at that facility. The underlying assumption is that higher facility popularity reflects greater attractiveness to visitors, moderated by the associated waiting time. The utility function U_x for path x with n nodes is defined as follows:

$$U_x = \frac{\sum_{f \in p_j} Pop(f)}{Q(p_x | p_{x-1})} \tag{2}$$

where $Q(p_x|p_{x-1})$ is the expected queuing time at path p_x given p_{x-1}, and $Pop(p_x)$ is the sum of popularity of all facilities in the path. The path's expected queuing time $Q(p_x|p_{x-1})$ is calculated by summing up the queuing time at all facilities:

$$Q(f_i) = \frac{1}{Cap(f_y)} Dur(f_y) \delta(f_{y,h}^{(x)} = f_{y,h}^{(x-1)}) \tag{3}$$

where $\delta(f_{y,h}^{(x)} = f_{y,h}^{(x-1)}) = 1$ if the facility appears to overlap between paths p_x and p_{x-1} within the same hour h. Capacity $Cap(f_x)$ is set to be a constant for simplicity. Finally, the transition matrix T is defined as:

$$T_{ij} = \frac{\sum_{f \in p_j} Pop(f)}{Q(p_j | p_{j-1=i})} \tag{4}$$

The transition matrix is then normalized by:

$$T_{ij} := \frac{T_{ij}}{\sum_j T_{ij}} \tag{5}$$

The set of feasible paths, i.e., total search space, is determined by solving an optimization problem as follows:

$$\begin{aligned} \text{maximize} \quad & TT_x = \sum_{i=1}^{n_x} Dur(f_i) + \sum_{j=1}^{n_x-1} Trav(c_{j,j+1}) \\ \text{subject to} \quad & Dist(c_{j,j+1}) \leq s, \;\; TT_x \leq t \end{aligned} \tag{6}$$

for n facilities in the path, with a constant time budget t.

Finally, we model the strategic itinerary recommendation problem as a social welfare optimization problem as follows:

$$\begin{aligned} \text{maximize} \quad & W = \sum_x U_x p_x \\ \text{subject to} \quad & \sum_x TT_x \le t, \quad x \in \{1, ..., n\} \end{aligned} \tag{7}$$

for n agents and time budget t.

4.3 Proof of NP-Hardness

We further investigate the NP-hardness of various sub-problems and show the respective proofs in this section.

Theorem 1 *The pathfinding problem defined in Equation 6 is NP-hard.*

Proof We prove the NP-hardness of the pathfinding problem by reducing from the 0-1 Knapsack problem, which is known to be NP-hard [61]. Recall that the 0-1 Knapsack problem is a decision problem as follows:

$$\begin{aligned} \text{maximize} \quad & z = \sum_i p_i x_i \\ \text{subject to} \quad & \sum_i w_i x_i \le c \\ & x_i \in \{0, 1\}, \quad i \in \{1, ..., n\} \end{aligned} \tag{8}$$

for n available items where x_i represents the decision of packing item i, p_i is the profit of packing item i, w_i is the weight of item i, c is the capacity of the knapsack.

Intuitively, the pathfinding problem is a decision problem of allocating a set of facilities into a path with a time budget constraint, where each facility has properties of a profit and a duration.

Formally, we transform the minimization problem in Equation 6 into an equivalent maximization problem. Concretely, the binary variable $f_i \in \{0, 1\}$ is included, where $f_i = 1$ if f_i is in path p_x, and 0 if otherwise. Furthermore, we define the profit of facility f_i as $p_i = -Dur(f_i)$ and set the travel time $Trav(c_{ij})$ to be a constant. Finally, the distance constant cap s is set to be infinity. The new problem formulation is represented as follows:

$$\begin{aligned} \text{maximize} \quad & T'_{path} = \sum_i p_i f_i \\ \text{subject to} \quad & \sum_i Dur(f_i) f_i \le t \\ & f_i \in \{0, 1\}, \quad i \in \{1, ..., n\} \end{aligned} \tag{9}$$

In this formulation, a path is equivalent to the knapsack in the 0-1 Knapsack problem, where each facility has its profit of p_i), and its cost of $Dur(f_i)$ that is equivalent to the profit and weight of an item respectively. The maximization problem is subjected to a constant time budget t, which is equivalent to the capacity c in a 0-1 Knapsack problem.

As a result, for any instance of the 0-1 Knapsack problem (i.e., item allocation decisions), we can find an equivalent instance of the pathfinding problem (i.e., facility allocation decision). Therefore, a solution in the pathfinding problem yields an equivalent solution to the 0-1 Knapsack decision problem. As such, we have completed the proof of NP-hardness for our path-finding problem to be NP-hard.

□

Theorem 2 *The social welfare optimization problem defined in Equation 7 is NP-hard.*

Proof Once again, we prove the NP-hardness of our welfare optimization problem by reducing it from the 0-1 Knapsack problem.

In Equation 7, the set of paths assigned to agents in the system is equivalent to the set of items in the 0-1 Knapsack problem; each path has its utility and total time, which are equivalent to the profit and weight of an item respectively; the maximization problem is subjected to a constant time budget t which is equivalent to the capacity c in a 0-1 Knapsack problem.

As a result, for any instance of the 0-1 Knapsack problem decisions, we can find an equivalent instance of a path assignment decision that yields a solution to the original Knapsack decision problem. As such, we conclude the proof of NP-hardness and have shown that our welfare recommendation problem is NP-hard.

□

Next, we describe our proposed SCAIR algorithm for solving this crowd-aware itinerary recommendation problem.

5 Strategic and Crowd-Aware Itinerary Recommendation (SCAIR)

In this section, we describe our proposed SCAIR algorithm, which comprises the main steps of finding feasible paths, generating a transition matrix, and simulating traveler visits.

5.1 Finding Feasible Paths

Algorithm 1 shows the pseudocode of our path-finding algorithm based on a breadth-first strategy. The input is a graph $G(F, C)$ that represents a theme park with the set of facilities F and connections C, time budget TT_{max}, and distance limit between two facilities $Dist_{max}$. This algorithm then generates and returns a collection of feasible paths, $Paths$, based on the provided input graph $G(F, C)$.

We iterate the collection of intermediate $Paths$ and call the $FindViableFacilities$ function to find viable facilities, where $f_{-1}^{(i)}$ is the last facility of the path, and $Dist_{max}$ is the maximum distance an agent wants to travel from one facility to another. We set the parameters of total time budget $T_{max} < 8hours$ and maximum allowed distance between two facilities $Dist_{max}(f_{current}, f_{next}) < 200m$. If there is

Algorithm 1: SCAIR - FindFeasiblePaths()

```
Data: $f_i \in F, c_{ij} \in C, TT_{max}, Dist_{max}, f_0$
Result: $Paths$: the set of feasible paths
begin
    $Paths = [[f_0]]$;
    while $True$ do
        for $path_i \in Paths$ do
            $VF = FindViableFacilities(f^{(i)}_{-1}, Dist_{max})$;
            if $len(VF) == 0$ then
                $path_x = path_i + [FindNextNearest(f^{(i)}_{(-1)})]$;
                if $TT_x < TT_{max}$ and $path_x \notin Paths$ then
                    $Paths+ = [path_x]$;
                    $Paths.pop(path_i)$
                end
            end
            foreach $vf \in VF$ do
                $path_x = path_i + [vf]$;
                if $TT_x < TT_{max}$ and $path_x \notin Paths$ then
                    $Paths+ = [path_x]$;
                end
            end
            $Paths.pop(path_i)$;
        end
        if $AllPathsMaxTimeBudget(Paths)$ or $AllPathsReachFullLength(Paths)$ then
            break;
        end
    end
end
```

no available facility that meets the distance constraint and the path has sufficient time budget remaining, the agent proceeds to the next nearest facility. We also do not allow an agent to revisit a facility on the same trip.

5.2 Transition Matrix

SCAIR introduces a discrete-time state encoder that acts as the transitional matrix of the Markov chain. The row and column headers are the POIs, and the cells record the expected transitional utilities from one POI to the other at any given time. The transitional utilities at a given time represent the expected utilities of the optimal path.

Specifically, we find the set of feasible paths (discussed in section 5.1) to construct a transition matrix T by calculating T_{ij} as the costs of taking path j given path $j-1=i$. The output of the $FindCost()$ function varies based on the arrival interval λ because it affects the expected arrival time for each facility at $path_j$, which leads to the different occurrences of overlapping facilities between $path_i$ and $path_j$.

Algorithm 2 shows the pseudo-code of our algorithm to calculate the transition matrix for the next state.

The input is the transition matrix at current state SE, the POI information f_i, costs c_ij and a set of limitations $limits$.

Line 2. The algorithm starts with a 2-dimensional array, where the row and column headers are the lists of POIs. We loop through the SE and update each cell.

Line 3. Find all feasible paths where the starting POI is f_i.

Line 4. Filter paths that the second POI is f_j.

Algorithm 2: TransitionMatrixNext()

```
Data: $SE_t, f_i, c_{ij}, limits$
Result: progress one step for the state encoder
begin
    foreach $SE[i, j]$ do
        $paths = FindFeasiblePaths(f_i, c_{ij}, ...limits, f_0 = f_i)$;
        $paths_j = paths[f_1 = f_j]$;
        $U_j = []$;
        foreach $path\ in\ path_j$ do
            $u_p = CalculateUtility(path)$;
            $U_j.append(u_p)$;
        end
        $SE[i, j] = max(U_j)$;
    end
end
```

Line 5 to 9. Calculate the utility of each path in $paths_j$.

Line 10. Update the cell $SE[i, j]$ with the maximum utility.

5.3 Simulation

Algorithm 3 shows an overview of the simulation procedure, which involves iterating through the visit data of theme parks $Parks$, a list of time budgets $TimeBudgets$, and an array of arrival intervals $ArrivalIntervals$.

Algorithm 3: SCAIR - Simulate()

```
Data: $Parks, TimeBudgets, ArrivalIntervals$
Result: Export simulation data to a CSV file
begin
    $Results = \{\}$;
    for $Park \in Parks$ do
        for $SimTime \in TimeBudgets$ do
            for $\lambda \in ArrivalIntervals$ do
                $Paths = FindFeasiblePaths(Park, SimTime)$;
                $T = ConstructTM(Park, Paths)$;
                $Qt, Pop, Utility = RunSimulation(Paths, \lambda, SimTime)$;
                $Update(Results, [Qt, Pop, Utility])$;
            end
        end
    end
    $ExportCsvFromDict(Results)$;
end
```

Line 2. The algorithm starts with constructing a 2-dimensional array, where each row represents a path as a sequence of facilities visited. We then conduct a breadth-first search (line 3 to 25), starting with the first row with an element of the initial facility, i.e., the entrance of a theme park.

Line 6 to 11. Suppose the algorithm is unable to find a facility within the feasible range. In that case, it will instead find the nearest facility that is not yet visited and assign the new path into the $Paths$ collection if two conditions are met, namely (1) the new path's total time is within the visitor's time budget TT_{max}, and (2) no identical path exists in the $Paths$ collection. Eventually, we remove the path the iteration started off.

Line 13 to 20. If the algorithm manages to find a set of viable facilities, it will then iterate through the set and execute a similar selection process.

Line 22 to 24. The algorithm breaks out from the infinite loop when any one of two conditions is met, namely (1) all paths in the $Paths$ collection have reached their time budget, i.e., any additional facility will make the total time of a path to be larger than the visitor's time budget; or (2) every path has included all available facilities.

Line 6 to line 13. For each step, the $FindFeasiblePaths()$ function finds the set of feasible paths which enables the $ConstructTM()$ function to construct the transition matrix, with input parameters namely park data $Park$ and simulation time $SimTime$. The $RunSimulation()$ function then simulates to find the total queuing time Qt, the average popularity among all facilities visited Pop, and the expected utility $Utility$ which is calculated as a function of Qt and Pop. Finally, after completing the simulations, we update the $Results$ dictionary (Line 9) and export the experimental data into CSV files (line 13).

6 SCAIRv2

In this section, we describe our proposed SCAIRv2 algorithm, which overcomes the limitation of SCAIR, where its complexity restricts it from running large-scale simulations.

6.1 Limitation of SCAIR

SCAIR has a factorial time complexity that grows with the number of facilities in a park and the number of visitors. It utilizes a greedy approach for the next-POI recommendation. SCAIR calculates the expected utility of the optimal path starting from the targeted POI, considering the crowd information and waiting time. While SCAIR can handle small-scale environments with time constraints, given a larger setting however, the search process may take too long to recommend an optimal path in time in practice.

6.2 Discrete-time State Encoding

To overcome the problem mentioned earlier, we propose the second version of SCAIR, or *SCAIRv2*, which introduces a discrete-time state encoder to record crowd distribution and recommend optimal paths in real-time. Specifically, a state encoder is a two-dimensional array. The row indexes are the time steps calculated by the operating hours divided by a preset interval in between time steps. For instance, suppose we simulate a theme park with 10 operating hours and set a 5-min interval. The size of the row indexes is $10 * 60/5 = 120$. The column indexes are the list of facilities' IDs. The value in each cell represents the number of visitors appearing in the respective POI at that time step.

Next, we discuss the implementation of the main algorithms for SCAIRv2.

6.3 Initialising the State Encoder

We initialize the state encoder with zero values and conduct one update step. Algorithm 4 shows an overview of the procedure to initialize the state encoder in SCAIRv2.

Line 2 to 5. The algorithm takes in the time-step interval $Interval$, max simulation time $MaxTime$, i.e., the operating hours of the simulated park, and the IDs

Algorithm 4: SCAIRv2 - InitSE()

```
Data: $Interval, MaxTime, PoiIDs$
Result: SE
begin
    $RowIdx = []$;
    for $idx \in range(1, MaxTime/Interval)$ do
        $RowIdx.append(idx)$
    end
    $SE = DataFrame(data = 0., index = RowIdx, columns = PoiIDs)$;
    $SE = UpdateSE(SE)$;
end
```

of all POIs $PoiIDs$. We use uniformly spaced time steps as row headers and the POIs as column headers to generate the state encoder.

Line 6 to 7. The algorithm creates the state encoder and initializes it with zeros in all cells. Subsequently, it calls algorithm 5 to update the initial state encoder with the first arrival. At the end of the algorithm, it outputs the initial state encoder.

6.4 Updating the State Encoder

Algorithm 5 describes the procedure of updating the state encoder for one time step. It takes in a state encoder SE, the current time step $CurrTimeStep$, and a path $Path$, and outputs the updated state encoder $SEUpdated$.

Algorithm 5: SCAIRv2 - UpdateSE()

```
Data: $SE, CurrTimeStep, Path$
Result: SEUpdated
begin
    $Step = CurrentTimeStep$;
    while $Path$ do
        $POI = Path.pop(0)$;
        while $POI.Dur$ do
            $SE[POI, Step] += 1$;
            $Step += 1$;
            $POI.Dur -= SE.Interval$;
        end
        $SE[POI, Step] -= 1$;
    end
end
```

Line 3. The procedure loops through every POI in the given path and updates the state encoder accordingly.

Line 5 to 10. We add one count as the person stays in the POI and reduce one when he or she leaves. Note that $POI.Dur$ is the total time duration the visitor spends in the POI, and $SE.Interval$ is the time step given when initializing the state encoder.

6.5 Finding Optimal Path

Algorithm 6 describes SCAIRv2's search algorithm of an optimal path at a given state. It requires inputs of a state encoder SE, an initial POI $InitialPOI$, and the maximum time constraint $MaxTime$, and outputs an optimal path $OptPath$.

Line 3 and 15. The algorithm iteratively finds the optimal path until it exhausts the $MaxTime$ limit.

Algorithm 6: SCAIRv2 - FindOptPath()

```
Data: $SE, InitialPOI, MaxTime$
Result: OptPath
begin
    $OptPath = []$;
    while (MaxTime ¿ 0) do
        $NextPOI = Null$;
        $OptDur = Null$;
        $OptUti = -1$;
        for POI in SE.ColumnIDs do
            $CurrPoiDur = Q(POI) + Dur(POI) + Trav(InitPOI, POI)$;
            $CurrPoiUti = Pop(POI)/CurrPoiDur$;
            if OptUti ¡ CurrPoiUti then
                $NextPOI = POI$;
                $OptDur = CurrPoiDur$;
                $OptUti = CurrPoiUti$;
            end
            $MaxTime- = OptDur$;
        end
        $OptPath.append(NextPOI)$;
    end
end
```

Line 8 and 9. The duration of the POI $CurrPoiDur$ is calculated by summing up the expected queuing time $Q(POI)$, the duration of visit $Dur(POI)$, and the travel time from initial POI to the target POI $Trav(InitPOI, POI)$. Note that the queuing time requires information on the capacity and the current crowd of the POI, which are not implicitly calculated in $Q(POI)$. The utility of the POI $CurrPoiUti$ is calculated by dividing the popularity of the POI $Pop(POI)$ by $CurrPoiDur$. Refer to section 4.2 for the definitions and explanations of $Q(POI)$, $Dur(POI)$, $Trav(InitPOI, POI)$ and $Pop(POI)$.

Line 17. Together with the POI IDs, we also append the properties of the POIs to simplify computation for other algorithms.

6.6 Algorithmic Complexity Analysis

The state encoder records the crowd distribution for all POIs at any given time. The encoder has a size of n by m, where n denodes the number of POIs and m denotes the time intervals.

Theorem 3 *The state encoding mechanism takes linear-time to update.*

Proof The state encoding updates the encoder once per arrival. Each update takes at most $n * m$ steps. As m is a predefined parameter, we have the time complexity of updating the state encoder as $O(n)$.

□

Theorem 4 *The pathfinding algorithm has a linear time complexity.*

Proof SCAIRv2 takes one step to find the optimal path leveraging the state encoder, which records the crowd distribution for all POIs. Specifically, SCAIRv2 scans through the state encoder of size n by m, where n denotes the number of POIs and m is a constant that denotes the number of time steps. Subsequently, for each of the selected POIs, SCAIRv2 updates the state encoder on the crowd distribution, which takes m steps.

Therefore, we conclude that the time complexity of pathfinding algorithm is:

$$n * m + m = O(n) \tag{10}$$

□

Theorem 5 *SCAIRv2 has a linear-time space complexity.*

Proof For space complexity, SCAIRv2 stores a 2D array with size n by m, where m is a constant. Therefore, the space complexity of SCAIRv2 is:

$$n * m = O(n) \tag{11}$$

□

7 Experiments

This section describes our dataset, evaluation process, and baselines.

7.1 Dataset

We conduct our experiments using a publicly available theme park dataset from [6]. This dataset is based on more than 655k geo-tagged photos from Flickr and is the first that includes the queuing time distribution of attractions in various Disney theme parks in the United States.

The dataset contains the POIs, user visits, distance, and popularity information of five theme parks data, namely Disney Hollywood Studios (DisHolly), EPCOT (Epcot), California Adventure (CalAdv), Magic Kingdom (MagicK), and Disney Land (Disland).

For SCAIRv1, we perform our experiments and evaluation using the dataset of user visits in Epcot and DisHolly, which contain 17 and 13 POIs, respectively.

As *SCAIRv2* substantially reduces the complexity of the earlier version of *SCAIR*, we evaluate it on two larger theme parks, namely, CalAdv and MagicK, in our experiments to demonstrate this aspect. These two parks have 25 and 27 POIs, respectively, which is 45% to 108% larger than DisHolly and Epcot.

While conducting the experiments, we realized some limitations in the original dataset. For instance, as it was released in 2017, some of the information, such as popularity, duration, and capacity, are not updated in a timely manner. Hence, some information provided in the dataset is not up to date, such as the capacity and duration of the facilities. This may be due to changes in the facilities over time.

To overcome the limitations, we leverage publicly available data sources, such as theme parks websites, Google Maps, and Wikipedia, to update the information dataset and add new features, such as average ratings and number of reviews, to the dataset to better reflect the popularity of the facilities. The newly updated dataset is publicly available at `https://github.com/junhua/SCAIR`.

7.2 Experimental Parameters

In our experiments, we empirically evaluate our proposed SCAIR algorithm across a comprehensive range of settings to thoroughly assess its performance robustness and scalability. The evaluation leverages two critical experimental parameters, which are the agent arrival interval λ and the simulation time budget T.

Specifically, the arrival interval λ defines the mean inter-arrival duration between consecutive agents entering the system, measured in minutes. We explored a broad range of λ values to reflect diverse real-world visitor arrival scenarios, varying from high-density conditions ($\lambda \in \{0.01, 0.02, \ldots, 0.09\}$) to relatively sparse arrival scenarios ($\lambda \in \{0.1, 0.2, \ldots, 1.0\}$). This range allows us to rigorously test the performance and generalizability of SCAIR under both highly congested and lightly trafficked contexts.

For the simulation time budget T, we considered durations ranging from short, constrained visits to extended, leisurely visits. Specifically, we varied the total available visitor time from 60 to 360 minutes, incremented at regular intervals of 30 minutes ($T \in \{60, 90, 120, \ldots, 360\}$). This variety of time budgets helps us validate the capability of the algorithm in managing itinerary recommendations under differing visitor time constraints.

These experimental configurations were carefully determined based on empirical insights and preliminary tests, ensuring that the evaluated scenarios align closely with realistic use-cases. All evaluations were conducted over multiple runs to capture variability, and the performance results presented in subsequent sections reflect averaged metrics over all λ and T settings, providing robust and representative insights into the performance characteristics of our proposed methodology.

7.3 Evaluation and Baselines

We compare our proposed SCAIR algorithm against three competitive and realistic baselines. The first two algorithms are based on visitors' intuitive strategies in real-life [2], while the third is a greedy algorithm used in [14]. In summary, the three baseline algorithms are:

1 Distance Optimization (denoted as $DisOp$) [2]. An iterative algorithm where agents always choose the facility with the shortest distance to the currently chosen one.

2 Popularity Optimization (denoted as $PopOp$) [2]. An iterative algorithm where agents always choose the next most popular facility within the specified distance constraint from the current facility.

3 Popularity over Distance Optimization (denoted as $PodOp$) [14]. An iterative greedy approach models utility as the popularity of the POI normalized by the distance from the current one and iteratively chose the POI with the highest utility.

Similar to many itinerary recommendation works [20, 6], we adopt the following evaluation metrics:

1 Average Popularity of Itinerary (denoted as $AvgPop$). Defined as the average popularity of all attractions recommended in the itineraries.

2 Expected Queuing Time per Visitor (denoted as $AvgQt$). Defined as the average queuing time that each visitor spends waiting for attractions in the recommended itinerary.

3 Expected Utility (denoted as Uty). Defined as the average utility score for all users based on the recommended itineraries.

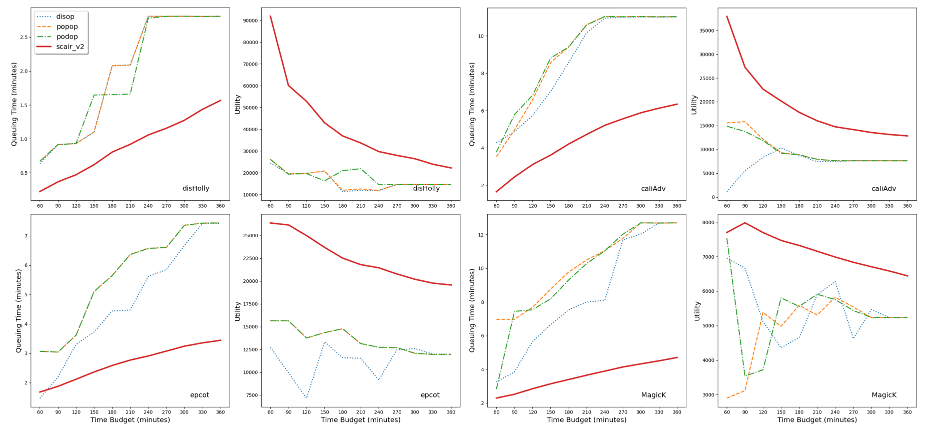

Figure 2 The plots show the performance of SCAIRv2 over four theme parks data. It shows that both of the queuing times and utility change with respect to simulation time T. We observe that SCAIRv2's queuing time is consistently and significantly lower than the baselines, and SCAIRv2's utility outperforms all the baselines in a majority of cases.

8 Results and Discussion

Figure 2 shows the experimental results of our proposed SCAIR algorithm compared to the three baseline algorithms. The x-axis indicates the time budget of visits, and the y-axis indicates the queuing time, popularity, and utility. To examine the effects of different user arrival frequencies, multiple experiments are conducted based on different arrival intervals λ, i.e., from 0.01 to 0.1 with a step size of 0.01 and from 0.1 to 1.0 with a step size of 0.1. The values in the graph are averaged across all λ.

Table 1 Queuing Time Ratio (Smaller values are better)

	Disney Hollywood (DisHolly)	Epcot Theme Park (Epcot)
DisOp	0.045 ± 0.221	0.076 ± 0.414
PopOp	0.046 ± 0.215	0.092 ± 0.368
PodOp	0.045 ± 0.211	0.092 ± 0.368
SCAIR	*0.003±0.010*	*0.016±0.006*

8.1 Queuing Time

In general, we observe that SCAIR outperforms the queuing time and utility baselines in the four theme parks. SCAIR is able to maintain a low queuing time with different time budgets, while the baseline's queuing time increases with the growth of the time budget. The observation is consistent for both theme parks. Table 1 shows the ratio of queuing time and time budget of visitors. SCAIR produces a queuing time ratio that is 78.9% to 93.4% shorter than that of the baselines across both DisHolly and Epcot theme parks.

8.2 Utility
For Utility, SCAIR outperforms all baselines consistently across all-time budgets for four theme parks. The main contributing factor to this result is the much improved queuing time performance that SCAIR can achieve compared to the various baselines. In turn, the reduced queuing time leads to a higher utility score as tourists can utilize more of their time budget in visiting attractions rather than spending excessive time queuing.

8.3 Limitations and Trade-offs
Our proposed SCAIR algorithm has been shown to perform well in terms of queuing time and utility in the general scenario. However, there are also various limitations that may affect its performance due to unique scenarios. Firstly, there is an implicit trade-off between a globally-fair utility versus personal utility, i.e., there may be individuals with their interest preferences being less aligned, although the global population benefits overall. Secondly, there may be the need for additional incentives to ensure that there is a global adherence or adoption of the recommended itineraries, which may result in additional costs for the facility/service providers. Lastly, different definitions of utility may be harder to model accurately, e.g., changing interest preferences over time, which may in turn affect the effectiveness of SCAIR.

9 Conclusion and Future Work
We now summarize the main findings of our work and discuss some possible directions for future research.

9.1 Conclusion and Discussion
Traditional itinerary recommendation methods and many prior works primarily target individual users, assuming isolated decision-making from the individual traveler perspective. However, this assumption breaks down in real-world environments, such as theme parks, where multiple users simultaneously follow similar recommendations. This leads to the well-known Selfish Routing problem, where agents pursuing individual utility inadvertently reduce overall system efficiency. For example, when all travelers are recommended the same POIs with a short queuing time based on historical data, those POIs then become congested and suffer from a long queuing time.

In this paper, we introduced the Strategic and Crowd-Aware Itinerary Recommendation (SCAIR) algorithm, which formulates the itinerary recommendation task as a social welfare optimization problem. By explicitly modeling crowd behavior and inter-agent interactions, SCAIR shifts from a user-centric to a system-level perspective. We address the limitations of static, single-agent models by leveraging Markov Decision Processes and introducing a novel state encoding mechanism that supports efficient real-time itinerary planning. Drawing from game-theoretic principles and Markov Decision Processes, SCAIR explicitly models the impact of agent interactions and crowd dynamics on route quality. This formulation not only aligns with foundational concepts such as the Price of Anarchy and Braess's Paradox, but also provides a tractable solution to these coordination failures in dynamic environments.

Our complexity analysis demonstrates that the enhanced algorithm, SCAIR, significantly reduces computational overhead, enabling linear-time updates and pathfinding. Experimental results across four real-world theme park datasets show that SCAIR consistently outperforms established baselines in both queuing time reduction and overall utility, validating the importance of incorporating crowd-aware mechanisms.

Importantly, the benefits of SCAIR remain robust across varying arrival intervals and time budgets, illustrating its adaptability to dynamic visitor patterns. Moreover, we updated and extended a public dataset to better support research on scalable, real-time itinerary recommendation systems. This enriched dataset, combined with our proposed algorithm, potentially opens up future research in socially optimal path planning.

9.2 Future Work

Building on the current work, there are several promising directions that could be explored.

Firstly, we plan to extend the current formulation into a multi-objective optimization framework. In contrast to optimizing a single utility function, we aim to separately model and balance multiple objectives, such as minimizing queuing time and maximizing attraction popularity, based on the principle of Pareto efficiency. This will allow us to uncover trade-offs between competing goals and provide more balanced itinerary recommendations that reflect varied user and system-level preferences. Additionally, we will explore alternative formulations of the utility function to better capture visitor satisfaction and system performance under different assumptions.

Secondly, to further improve simulation efficiency and scalability, we will explore advanced metaheuristic and scheduling algorithms from related domains. These include but are not limited to the Adaptive Genetic Algorithm[24], Tabu Search[25], Variable Neighborhood Search (VNS)[26], Simulated Annealing[27], and Branch-and-Cut methods [28]. Integrating these solvers could improve runtime and solution quality in larger and more complex theme park environments.

Thirdly, we see value in extending our strategic recommendation framework to other game-theoretic applications, such as knowledge acquisition [62, 63], crisis management [39, 64], and career path planning [65, 66]. These domains also involve multi-agent coordination and resource constraints, making them a natural adaptation of our proposed algorithm for strategic, crowd-aware recommendation models.

Lastly, we intend to develop more realistic behavioral models to simulate visitor decision-making. Future versions of our simulator will incorporate park-specific topology features, such as entrance and exit locations, to more accurately represent spatial constraints. We also plan to explore probabilistic path choice models, using softmax-based selections instead of deterministic one-hot assignments, to better represent the stochastic and uncertain nature of human navigation.

Ethics approval and consent to participate
Not applicable.

Consent for publication
The authors consent to the publication of this manuscript.

Availability of data and materials
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Funding
This research is funded by MOE-T2EP20123-0015.

Author's contributions
JL designed the research, ran experiments, analyzed results, and wrote the manuscript. AG, KLW and KHL designed the research, analyzed results and contributed to manuscript preparation. All authors read and approved the final manuscript.

Acknowledgements
This research is supported by the Ministry of Education, Singapore, under its Academic Research Fund Tier 2 (Award No. MOE-T2EP20123-0015). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of the Ministry of Education, Singapore.

Author details
[1]Singapore University of Technology and Design, Singapore. [2]Singapore Management University, Singapore. [3]University of Colorado Denver, USA.

References

1. Zhang, C., Liang, H., Wang, K.: Trip recommendation meets real-world constraints: Poi availability, diversity, and traveling time uncertainty. ACM TOIS **35**(1), 5 (2016)
2. Lim, K.H., Chan, J., Leckie, C., Karunasekera, S.: Personalized trip recommendation for tourists based on user interests, points of interest visit durations and visit recency. Knowledge and Information Systems **54**(2), 375–406 (2018)
3. Choudhury, M.D., Feldman, M., Amer-Yahia, S., Golbandi, N., Lempel, R., Yu, C.: Automatic construction of travel itineraries using social breadcrumbs. In: Proceedings of HT'10, pp. 35–44 (2010)
4. Gionis, A., Lappas, T., Pelechrinis, K., Terzi, E.: Customized tour recommendations in urban areas. In: Proceedings of WSDM'14, pp. 313–322 (2014)
5. Padia, P., Lim, K.H., Chan, J., Harwood, A.: Sentiment-Aware and Personalized Tour Recommendation. In: Proceedings of BigData (2019)
6. Lim, K.H., Chan, J., Karunasekera, S., Leckie, C.: Personalized itinerary recommendation with queuing time awareness. In: Proceedings of SIGIR'17, pp. 325–334 (2017). ACM
7. De Choudhury, M., Feldman, M., Amer-Yahia, S., Golbandi, N., Lempel, R., Yu, C.: Automatic construction of travel itineraries using social breadcrumbs. In: Proceedings of the 21st ACM Conference on Hypertext and Hypermedia, pp. 35–44 (2010). ACM
8. Chen, L., Cao, J., Chen, H., Liang, W., Tao, H., Zhu, G.: Attentive multi-task learning for group itinerary recommendation. Knowledge and Information Systems **63**(7), 1687–1716 (2021)
9. Sarkar, J.L., Majumder, A.: gtour: Multiple itinerary recommendation engine for group of tourists. Expert Systems with Applications **191**, 116190 (2022)
10. Roughgarden, T.: Selfish Routing and the Price of Anarchy vol. 174. MIT press Cambridge, ??? (2005)
11. Osborne, M.J., Rubinstein, A.: A Course in Game Theory. MIT press, ??? (1994)
12. Liu, J., Wood, K.L., Lim, K.H.: Strategic and crowd-aware itinerary recommendation. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 69–85 (2020). Springer
13. Halder, S., Lim, K.H., Chan, J., Zhang, X.: Efficient itinerary recommendation via personalized poi selection and pruning. Knowledge and Information Systems **64**(4), 963–993 (2022)
14. Zhang, C., Liang, H., Wang, K., Sun, J.: Personalized trip recommendation with poi availability and uncertain traveling time. In: Proceedings of CIKM'15, pp. 911–920 (2015)
15. Lim, K.H., Wang, X., Chan, J., Karunasekera, S., Leckie, C., Chen, Y., Tan, C.L., Gao, F.Q., Wee, T.K.: PersTour: A personalized tour recommendation and planning system. In: Extended Proceedings of HT'16 (2016)
16. Wang, X., Leckie, C., Chan, J., Lim, K.H., Vaithianathan, T.: Improving personalized trip recommendation to avoid crowds using pedestrian sensor data. In: Proceedings of CIKM'16, pp. 25–34 (2016)
17. Ataei, M., Divsalar, A., Saberi, M.: The bi-objective orienteering problem with hotel selection: an integrated text mining optimisation approach. Information Technology and Management, 1–29 (2022)
18. Gama, R., Fernandes, H.L.: A reinforcement learning approach to the orienteering problem with time windows. Computers & Operations Research **133**, 105357 (2021)
19. Khodadadian, M., Divsalar, A., Verbeeck, C., Gunawan, A., Vansteenwegen, P.: Time dependent orienteering problem with time windows and service time dependent profits. Computers & Operations Research **143**, 105794 (2022)
20. Lim, K.H., Chan, J., Karunasekera, S., Leckie, C.: Tour Recommendation and Trip Planning using Location-based Social Media: A Survey. Knowledge and Information Systems **60**(3), 1247–1275 (2019)
21. Gunawan, A., Lau, H.C., Vansteenwegen, P.: Orienteering problem: A survey of recent variants, solution approaches and applications. European Journal of Operational Research **255**(2), 315–332 (2016). doi:10.1016/j.ejor.2016.04.059
22. Gunawan, A., Lau, H.C., Vansteenwegen, P.: Orienteering problem: A survey of recent variants, solution approaches and applications. European Journal of Operational Research **255**(2), 315–332 (2016)
23. Vansteenwegen, P., Souffriau, W., Oudheusden, D.V.: The orienteering problem: A survey. Euro. J. of Operational Rsch. **209**(1), 1–10 (2011)
24. Han, S., Xiao, L.: An improved adaptive genetic algorithm. In: SHS Web of Conferences, vol. 140, p. 01044 (2022). EDP Sciences

25. Li, M., Hao, J.-K., Wu, Q.: Learning-driven feasible and infeasible tabu search for airport gate assignment. European Journal of Operational Research **302**(1), 172–186 (2022)
26. Cai, J., Lu, S., Cheng, J., Wang, L., Gao, Y., Tan, T.: Collaborative variable neighborhood search for multi-objective distributed scheduling in two-stage hybrid flow shop with sequence-dependent setup times. Scientific Reports **12**(1), 1–19 (2022)
27. Venkateswaran, C., Ramachandran, M., Ramu, K., Prasanth, V., Mathivanan, G.: Application of simulated annealing in various field. Materials and its Characterization **1**(1), 01–08 (2022)
28. Lam, E., Le Bodic, P., Harabor, D., Stuckey, P.J.: Branch-and-cut-and-price for multi-agent path finding. Computers & Operations Research **144**, 105809 (2022)
29. Zhang, Y., Tang, J.: Itinerary planning with time budget for risk-averse travelers. European Journal of Operational Research **267**(1), 288–303 (2018)
30. Xiao, G., Tong, H., Shu, Y., Ni, A.: Spatial-temporal load prediction of electric bus charging station based on s2tat. International Journal of Electrical Power & Energy Systems **164**, 110446 (2025)
31. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
32. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (2019). doi:10.18653/v1/N19-1423. https://www.aclweb.org/anthology/N19-1423
33. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI Blog **1**(8) (2019)
34. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: Xlnet: Generalized autoregressive pretraining for language understanding. In: Advances in Neural Information Processing Systems, pp. 5754–5764 (2019)
35. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research **21**(140), 1–67 (2020)
36. Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H.W., Sutton, C., Gehrmann, S., et al.: Palm: Scaling language modeling with pathways. arXiv preprint arXiv:2204.02311 (2022)
37. Lee, L.-H., Lu, Y., Chen, P.-H., Lee, P.-L., Shyu, K.-K.: Ncuee at mediqa 2019: Medical text inference using ensemble bert-bilstm-attention model. In: Proceedings of the 18th BioNLP Workshop and Shared Task, pp. 528–532 (2019)
38. Singhal, T., Liu, J., Mu, W., Blessing, L.T., Lim, K.H.: Photozilla: An image dataset of photography styles and its application to visual embedding and style detection. In: Proceedings of the International Conference on Advances in Social Networks Analysis and Mining, pp. 445–449 (2023)
39. Liu, J., Singhal, T., Blessing, L.T.M., Wood, K.L., Lim, K.H.: Epic30m: An epidemics corpus of over 30 million relevant tweets. In: Proceedings of the 2020 IEEE International Conference on Big Data (2020)
40. Liu, J., Ng, Y.C., Gui, Z., Singhal, T., Blessing, L., Wood, K.L., Lim, K.H.: Title2vec: a contextual job title embedding for occupational named entity recognition and other applications. Journal of Big Data **9**(1), 1–16 (2022)
41. Hong, M., Jung, J.J.: Multi-criteria tensor model for tourism recommender systems. Expert Systems with Applications **170**, 114537 (2021)
42. Noorian, A., Harounabadi, A., Ravanmehr, R.: A novel sequence-aware personalized recommendation system based on multidimensional information. Expert Systems with Applications **202**, 117079 (2022)
43. Praditya, N.W.P.Y., Permanasari, A.E., Hidayah, I.: Designing a tourism recommendation system using a hybrid method (collaborative filtering and content-based filtering). In: 2021 IEEE International Conference on Communication, Networks and Satellite (COMNETSAT), pp. 298–305 (2021). IEEE
44. Hamid, R.A., Albahri, A.S., Alwan, J.K., Al-Qaysi, Z., Albahri, O.S., Zaidan, A., Alnoor, A., Alamoodi, A.H., Zaidan, B.: How smart is e-tourism? a systematic review of smart tourism recommendation system applying data management. Computer Science Review **39**, 100337 (2021)
45. Zhou, X., Tian, J., Peng, J., Su, M.: A smart tourism recommendation algorithm based on cellular geospatial clustering and multivariate weighted collaborative filtering. ISPRS International Journal of Geo-Information **10**(9), 628 (2021)
46. Xu, C., Liu, D., Mei, X.: Exploring an efficient poi recommendation model based on user characteristics and spatial-temporal factors. Mathematics **9**(21), 2673 (2021)
47. Halder, S., Lim, K.H., Chan, J., Zhang, X.: Poi recommendation with queuing time and user interest awareness. Data Mining and Knowledge Discovery, 1–31 (2022)
48. Yang, S., Liu, J., Zhao, K.: Getnext: trajectory flow map enhanced transformer for next poi recommendation. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1144–1153 (2022)
49. Garcia, I., Sebastia, L., Onaindia, E.: On the design of individual and group recommender systems for tourism. Expert Systems with Applications **38**(6), 7683–7692 (2011)
50. Anagnostopoulos, A., Atassi, R., Becchetti, L., Fazzone, A., Silvestri, F.: Tour recommendation for groups. Data Mining and Knowledge Discovery **31**(5), 1157–1188 (2017)
51. Hu, F., Huang, X., Gao, X., Chen, G.: Agree: Attention-based tour group recommendation with multi-modal data. In: Proceedings of DASFAA'19, pp. 314–318 (2019)
52. Gunawan, A., Yuan, Z., Lau, H.C.: A mathematical model and metaheuristics for time dependent orienteering problem. In: Proceedings of PATAT'14, pp. 202–217 (2014)
53. Koutsoupias, E., Papadimitriou, C.: Worst-case equilibria. In: Proceedings of STACS'99, pp. 404–413 (1999)
54. Papadimitriou, C.H.: Algorithms, games, and the internet. In: ICALP'01, pp. 1–3 (2001)

55. Braess, D.: Über ein paradoxon aus der verkehrsplanung, pp. 0042–0573. Physica-Verlag, ??? (1968). https://doi.org/10.1007/BF01918335
56. Pas, E.I., Principio, S.L.: Braess' paradox: Some new insights, pp. 265–276 (1997)
57. Piliouras, G., Nikolova, E., Shamma, J.S.: Risk sensitivity of price of anarchy under uncertainty. ACM Trans. Econ. Comput. **5**(1) (2016)
58. Nguyen, T.T., Roos, M., Rothe, J.: A survey of approximability and inapproximability results for social welfare optimization in multiagent resource allocation. Annals of Mathematics and Artificial Intelligence **68**(1-3), 65–90 (2013)
59. Papadimitriou, C.H., Tsitsiklis, J.N.: The complexity of markov decision processes. Mathematics of operations research **12**(3), 441–450 (1987)
60. Arora, S., Barak, B.: Computational Complexity: a Modern Approach. Cambridge University Press, ??? (2009)
61. Martello, S., Pisinger, D., Toth, P.: Dynamic programming and strong bounds for the 0-1 knapsack problem. Management Science **45**(3), 414–424 (1999)
62. Al-Gharaibeh, R.S., Ali, M.Z.: Knowledge sharing framework: A game-theoretic approach. Journal of the Knowledge Economy **13**(1), 332–366 (2022)
63. Deuser, K., Naumov, P.: Strategic knowledge acquisition. ACM Transactions on Computational Logic (TOCL) **22**(3), 1–18 (2021)
64. Seaberg, D., Devine, L., Zhuang, J.: A review of game theory applications in natural disaster management research. Natural Hazards **89**(3), 1461–1483 (2017)
65. Dave, V.S., Zhang, B., Al Hasan, M., AlJadda, K., Korayem, M.: A combined representation learning approach for better job and skill recommendation. In: Proceedings of CIKM'18, pp. 1997–2005 (2018). ACM
66. Hancock, S.: A future in the knowledge economy? analysing the career strategies of doctoral scientists through the principles of game theory. Higher Education **78**(1), 33–49 (2019)