What Properties do I have time to visit?

Christopher John Bond
School of Computing Technologies
RMIT University
Melbourne, Australia
s3767838@student.rmit.edu.au

Maria Spichkova
School of Computing Technologies
RMIT University
Melbourne, Australia
maria.spichkova@rmit.edu.au

***Index Terms*—Scheduling, Property Inspection Routing Planning**

***Abstract*—To find a place to rent or buy in a large city like Melbourne in Australia or Munich in Germany is a complicated quest. Searchers visit many flats, which require careful selection and planning of what locations can be visited, what visit duration should be set aside, and what kind of transport can be used. Real estate portals and transport routing services don't provide any solution to this problem. In the proposed project, we aim to provide a solution to this problem domain. Further, we will assess whether existing journey planning algorithms can be applied to the problem, suggest optimisations and determine if the algorithm delivers a result within an appropriate amount of time. In this project proposal, we analyse the differences in property inspection route planning to related research domains and present a preliminary specification in Section II. Section III introduces related work, and indicates the neighbouring problem domains whose solutions merit exploration, as they solve key aspects of this problem. It is concluded that the Orienteering Problem with Time Windows hosts the most promising algorithms, to be augmented with either Time Dependency attributes or use of a directions API to account for variable travel time and public transport. Section IV indicates the project's timeline for addressing the research questions defined below in Section I, aiming at delivering an evaluated model by 20-Sep-21 and a Minor Thesis by 28-Sep-21. The selected evaluation methodology is discussed in Section VI, which focuses on proving the model works rather than performance benchmarking due to its novel domain of application. Finally the outcomes expected from the Project are outlined in Section VII, which indicates a relatively faster algorithm run-time, potential concerns of map directions API use and the introduction of parking delay to driving estimation.**

I. Introduction

Australian Bureau of of Statistics (ABS) data shows that over 3.5 million Australian residents moved home in 2016, see [3]. Given the significance of the decision, most will not rely on online Multiple Listing Services (MLS) [also known as online real-estate listing service] pictures alone. MLS provide assistance with recommendation, but they don't provide any routing and customisation functionality.

Significant decisions, such as real estate inspection planning, require effortful decision making. This can result in fatigue, which leads to poor trade-offs, reduced decision accuracy, lowered satisfaction and increased regret, see [21]. Minimising the complexity of property inspection planning would positively impact the 3.5 million Australian's looking for an abode each year [3].

Let us illustrate the problem with a scenario: James Smith is looking for properties in Melbourne's inner west, he's selected 40 that are of interest, 36 of which have inspections on the upcoming Saturday. James knows which properties he prefers over the others, he generally takes about 15 minutes inspecting a property and he rathers driving over public transport. James can't work out which properties to select to get to the most inspections he prefers. Figure 1 illustrates a manual solution to this problem. The bar lengths depict duration. Each yellow bar depicts travel between property locations. Travel time from the previous primary property is situated immediately above the respective property. Each thin grey bar indicates the inspection window for a given property. Here we use the following visual notation:

- red bars - the properties selected for the visit, as result of the analysis of the provided data;
- blue bars - these properties were not selected for the visit, but they can be used for an alternative path as a "second best" choice based of the analysis of the provided data;
- grey bars - these properties were not selected for the visit, as their inspection times will close before a complete inspection. They they can be used only as a part of an alternative path that isn't optimal in terms of missing the opportunity to visit properties with higher priority, etc.

The words to the right are location names, and numbers to the left indicate ranked preference. Travel time from previous secondary properties is not represented in this graphic for ease of explanation, but is explored further in Figure 1.

The aim of our work is to develop an algorithm that solves James' problem, by generating a user-customisable route based on ranked preferences, geographic location of properties, the available inspection times, and preferred inspection duration.

A. Research Question

We will answer the following research questions:

RQ1: What parameters are attributes of Property Inspections? And therefore, how do Property Inspection Route Planning (PIRP) parameters differ from existing route planning problems?

RQ2: Can an existing journey planning algorithm be adjusted to meet the parameters defined in the PIRP problem? If yes, then how exactly should it be adjusted? If no, what algorithm will solve this problem?

RQ3: Is the performance of the proposed algorithm within the order of magnitude of online listing service recommendation algorithms?

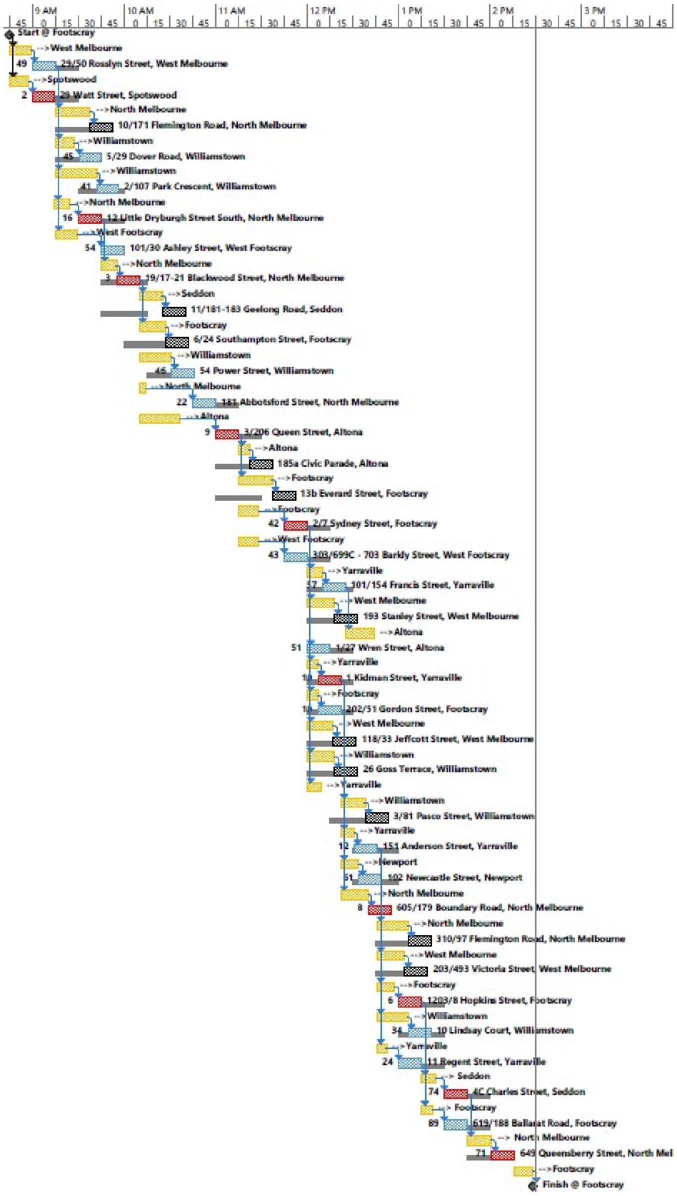

Fig. 1: Manually Determined Inspection Solution

II. Preliminary specification of PIRP

In PIRP, we will need at least the following parameters:

- Geographic locations of properties of interest,
- Inspection time of properties of interest,
- Preference rank per property,
- Standard inspection duration,
- The day the plan should be specified for.

The formal specification of PIRP, which addresses *RQ1*, will be elaborated during test case creation, finalising by 02-Jul-21.

Based on the property of interest, the property location and inspection window details will be retrieved from the online listing service. The user will receive:

- A selection of inspection to attend
- A list of alternative inspections than can be substituted for primary inspections

Below in Figure 2 is an example of user adjustment, based on the information in Figure 1. Here the user has clicked on *29/50 Rosslyn Street, West Melbourne* (row 3) instead of the algorithm selected *29 Watt Street, Spotswood* (row 5) [selected in Figure 1]. This means that James can't make it to *5/29 Dover Road, Williamstown* (row 9) because of the longer travel time, but now he can make it to *10/171 Flemington Road, North Melbourne* (row 7) because of reduced travel time. As we discuss in the user experience subsection of Section III, user adjustment is essential because users do not display consistency between what they project they will prefer and their actual preferences. Further, the standard weighting methodology applied to property rankings will imperfectly reflect user's preference. The approach described here maximises user choice, enabling users to adjust their plan, it also allows them to appreciate trade-offs in real time, without being overwhelmed by combinatorial complexity. The inspection process is already overwhelming, users like James would benefit from reduced complexity, minimising the symptoms of effortful decision making, covered in Section I.

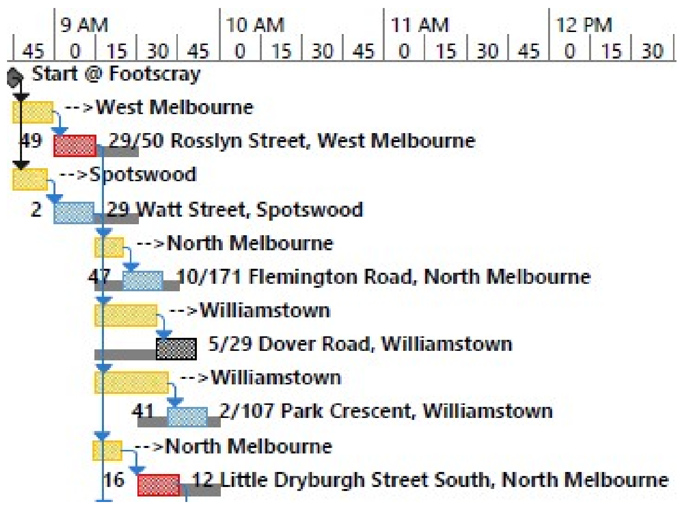

Fig. 2: User Alteration to Manual Solution

III. Literature Review

Routing problems are a mature area of research, where real-world applications have led to new domains of research. The core problem refers to maximising value and limiting cost in the context of multiple locations (referred to as nodes) with paths that connect them (edges). Each location offers a value, and each path a cost [31].

The original problem is the well-known Travelling Salesman Problem, which attempts to visit every location while minimising travel cost, see [4]. An evolution of this idea is referred to as the Travelling Salesman Problem with Profits (TSPP), it introduces varying value for each location, and eliminate the requirement to visit every location. TSPP retains the dual goals of maximising value, while minimising cost, see [11].

This concept spawned the Orienteering Problem (OP) [31], which considers the travel budget as a fixed roof, and the Vehicle Routing Problem (VRP), see [2], which introduces parameters of road navigation. These problems have established variants that consider time windows, including the (Team) Orienteering Problem with Time Windows [15] and the Technician Routing and Scheduling Problem (TRSP) [22] respectively. Time Windows are start and finish times within which the visit must be made to qualify as a valid visit. These problems solve the issue of scheduled technician appointments and point of interest opening-hours. They tend towards larger windows and shorter engagement time, resulting in several ordering options for the route.

Unfortunately, exact solutions in these fields are stated to be unrealistic for real-world application beyond a few properties (locations), due to high-order polynomial behaviour, see [13], [16], [22], [27]. Therefore, solutions focus on approximation algorithms that solve in lower-order polynomial time.

A. Approach

The literature search used IEEE Xplore, Science Direct and Google Scholar databases. Science Direct and IEEE Xplore provided more reliable sources, but yielded a lower volume.

Keywords used: *Travelling salesman problem with profits, Tourist Trip Design Problem, Orienteering Problem, Time Windows, Team Orienteering Problem and Directions API.*

B. Orienteering Problem

The Orienteering problem (OP), introduced by Tsiligirides, varies from the travelling salesman problem by converting the travel budget into a fixed constraint, where the Travelling Salesman Problem seeks to minimise travel cost, see [28]. Conceptually, Gunawan et. al. describe the OP as a combination of the classical Knapsack and Travelling Salesman Problems [15], which suggests that algorithms that perform well for solving both problems should also solve the combined problem well, a class of algorithms regularly applied in both domains are evolutionary algorithms, as presented in [1], [10], [26], [27], [30], [31].

The orienteering problem has variants for time windows (OPTW) for Point of Interest open-close times and time dependency (TDOP) for public transport usage. The orienteering problem represented in Figure 3, is an edge-weighted graph where preference is represented by radius of each circle, the selected path is made up of orange property locations (nodes), arrow paths (edges) [travel time length] and green start and end locations. Further, available properties, not selected are represented in blue. Based on [13]. Leading algorithms that directly solve this problem are Discrete Strengthened Particle Swarm Optimisation [26] and Greedy Randomized Adaptive Search Procedure and Path Relinking [5]. While these algorithms efficiently solve OP, they ignore time windows. This means a fundamental shift in algorithm construction would be required to make them relevant for our shortlist.

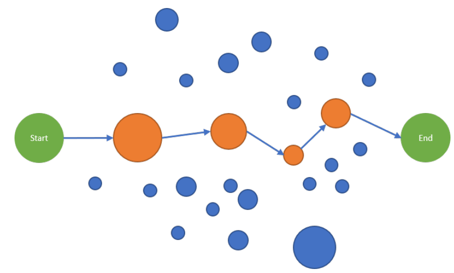

Fig. 3: Single tour Orienteering Problem

The tourist trip design sub-problem includes 3 elements: *Recommendation, Route Generation and Customisation Functions* [13]. The customisation functions focus on adding or removing points of interest. Similarly in our research in addition to route generation, we seek to provide customisation, allowing users to substitute selected properties with alternatives. Property recommendation is mature functionality for MLS, see [29], however integration with recommendation services will constitute future work.

As introduced earlier the tourist trip design sub-problem focusses on the OPTW variant, which has high relevance to the PIRP problem, given start and finish times are used. Iterated Local Search (ILS) was presented as a fast approach to solving OPTW problems, see [20], while a hybrid Simulated Annealing and Iterated Local Search (SAILS) algorithm presented as a more accurate, albeit computationally intensive solution approach, with a 5.19% score gap of an optimal solution within 1.5 seconds, see [16].

C. Team Orienteering Problem

The Team orienteering problem (TOP) enables multi-day [13] or multi-agent [16] scheduling by extending the single-path approach of OP into multiple paths, which was first introduced by Chao et. al. [6] . A pareto mimic algorithm was introduced by Ke et al. [17] that solves TOP, additionally Li et al. [18] put forward a dynamic programming algorithm and Verbeeck et al. [30] submitted an Ant Colony System (ACS) for Time Dependent Team Orienteering Problem (TD-TOP). These algorithms miss the fundamental Time Window parameter that is required by PIRP. However, if the PIRP solution internalises travel time calculation (as opposed to utilising a Directions API), time dependency will be an important component. Even more relevant are the adapted genetic algorithm submitted by Abbaspour et al. [1] and Ant Colony Optimisation (ACO) algorithm created by Verbeeck et al. [31], which address Time Dependence and Time Windows.

D. Vehicle Routing Problems

a) Technician Routing Problem: Mathlouthi et. al. [22] present an advanced algorithm within Technician routing that addresses a large number of attributes, several that are relevant to PIRP, for example time windows. However, many attributes are not relevant, for example skills, overtime, food breaks, tool and inventory use and equipment depots. The authors do not address algorithmic complexity, however the growth rate in wall-time from 20+ nodes (properties), suggests quadratic complexity.

b) Time Dependent Vehicle Routing Problem: Gmira et al. [14] present a Tabu Search algorithm that solves the Time Dependent Vehicle Routing Problem, which does not address the time window attributes of the PIRP problem.

c) Vehicle Routing Problem with Time Windows: Shen et al. [27] describe a hybridised algorithm called Ant Colony System with Brain Storm Optimisation, which addresses the Vehicle Routing Problem with time windows. Each of the above algorithms aim to minimise overall distance travelled, which makes their aim fundamentally different to both the PIRP and OP domains which seek to set overall time as a maximum value. This undermines their applicability to the PIRP problem due to the foundational changes required.

E. API use

Despite the wide-scale availability of directions APIs [25], leading routing algorithms rarely leverage them, and in some cases resort to crow-flight distance, and average-speed approximations to solve their problems, see [27]. An absence of API use in the OP and VRP literature indicates a risk that map APIs use may introduce delays, complexity or reduce model accuracy. Therefore, strategies to limit API calls will be considered. Incorporating APIs have been considered to reduce the scope of the PIRP solution and make it more real-world applicable. Similarly to our case, Zhu and Gonder [34] use map directions APIs to create alternative estimated routes with location pairs and durations, within the domain of cycle detection. This approach shows promise for routing in the PIRP problem and potential for future work for live navigation. Time Dependency would not be used in conjunction with an incorporated API because of their significant overlap in function. A Directions API will be tested for applicability for incorporation into the PIRP algorithm.

F. User Experience

Chritianto et al. [7] discussed the importance of presentation of choice to users in transportation solutions, further they indicate simple visualisation approaches are far more effective than descriptive approaches. As a result, this project will seek to present alternate routes in a simple box-based graphical form, similar to Figures 1 and 2, to enable simple visualisation and easy user customisation. Unfortunately, other transportation UX research located is rather dated, and therefore will not be leveraged, see [12].

Nielsen [24] indicates that user's stated preferences are not reliably consistent with their true preference. Therefore, it is important to allow for this variability in preference-oriented systems by allowing user-adjustment of solutions. Mueller et al. [23] indicate that best to worst scaling (comparably ranking each property) as compared with hedonic scaling (individual scale e.g. strongly dislike, dislike, like etc.) in preference measurement provides similar results which benefits the user by requiring them to answer with significantly less detail. However requires a trade off of a reduced level of detail of how closely grouped similarly preferred properties.

G. Recommendation Engine

All the algorithms explored above treat property(location)-weighting as a necessary process but out of scope for their algorithms. Given this gap, optimal list curation approaches were investigated. Yuan et. al. leverage image metadata to recommend routes based on time data, see [32], Lim et. al. [19] agreed and further utilised the data to recommend visit durations. It was found that MLS provide recommendation services that adequately address the recommendation function, with a high degree of maturity. Further the structure of MLS's encourages users to save preferred properties to *customisable lists*, see [29]. For the purpose of this project we will assume these lists can be ranked with relative ease, and this will be used as an input to the PIRP algorithm.

IV. PROJECT PLAN

The Key sections of the project are:

- Research Proposal
- Model Creation
- Evaluation
- Thesis Writing

A. Research Proposal

The research proposal submission and presentation time-frame and scope are defined within the scope of the Semester 1 2021 Research Methods course. Post-feedback, the proposal will be improved. This is important to conduct early, as it becomes much more challenging to adjust scope as the timeline matures, see [9]. Once this is complete it will be used as a basis for the Minor Thesis document skeleton.

B. Model Creation

After submission, the test cases and execution environment setup will be the next focus.

a) Test Cases: Test Cases that cover the core criteria in the evaluation section will be written, they will then be verified by the research supervisor as achievable and checked for bias. The writing of the test cases before model creation will reduce bias arising from solution thinking.

b) Determine Execution Environment: For the methodology section, the environment configuration will be described, enabling reproducibility of the experiments. This task will identify an environment (ideally public cloud) and the necessary configuration to efficiently run the experiment.

c) Modelling: As a starting point, leading models will be adopted (or as faithfully as reasonable, recreated) and verified to see if they can satisfy the PIRP use case without significant modification. This means that irrelevant constraints will be removed and any missing constraints will be added following the program architecture. No significant development can be conducted for the model to be considered 'modified'. If significant development occurs, the program will be considered

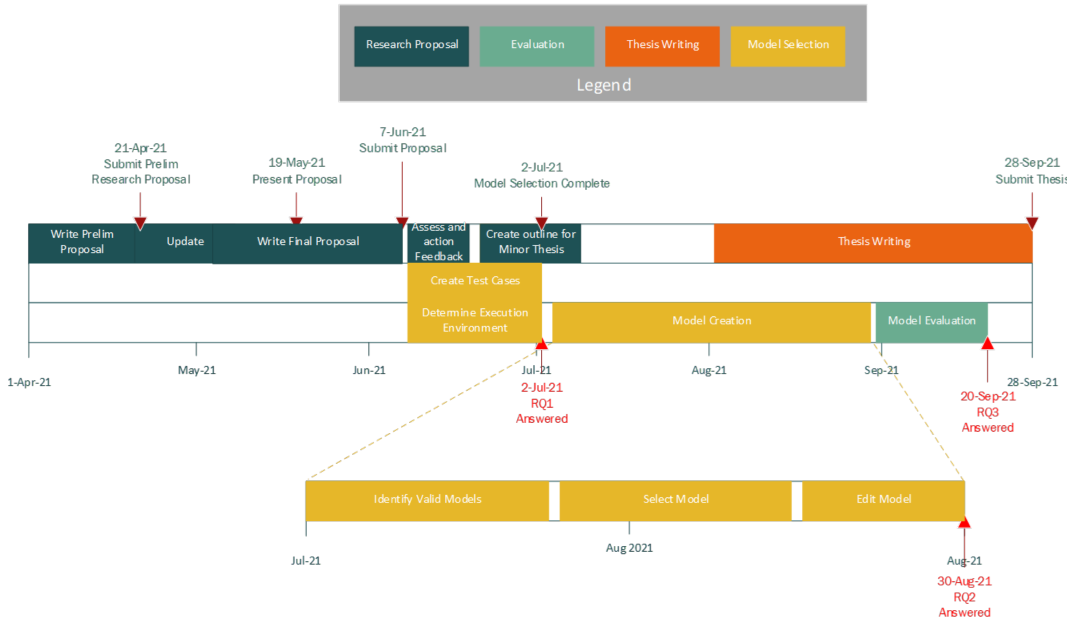

Fig. 4: Thesis Timeline

a new model. At this point, refinements will be investigated and iteratively applied. Alternatively, if no model adequately addresses the PIRP constraints, a new model will be created from first-principles, with guidance from existing models for architecture. Tests and refinement will be conducted iteratively to produce an evaluable algorithm. Once working, optimisation will be explored, for example, parallel computing, identified in the literature as being in its infancy in the OP domain, see [13].

The model will be broken into two parts. Part 1 will collect travel times (likely with the aid of a directions API). Part 2 will take the collected travel times and preferenced properties and conduct evaluation to deliver a modifiable recommended route and alternatives to the user.

C. Evaluation

On completion of the modelling phase, using a held-back data set, the model will be evaluated for performance and correctness. These factors are discussed in detail in the Evaluation section.

D. Thesis Writing

The thesis will have elements contributed to it throughout the timeline, the final results and discussions will require creation and validation phase outputs. Thesis writing will be conducted in earnest through mid-August to the submission deadline (28 September 2021).

V. Preliminary Analysis

Mathlouthi et. al. indicate that an increase in the duration-taken:time-window ratio (as in PIRP compared with classic OPTW) reduces combinatorial complexity. This reduction is due to reduced flexibility of properties (locations) with available time windows, see [22]. This bodes well for the complexity of the PIRP algorithm which demonstrates a median of 30 minute time windows in an informal study of over 1000 MLS inspection durations in Sydney and Melbourne. It is predicted that users will select inspection durations ranging from 10-30 minutes, presenting a high duration-taken:time-window ratio.

Additionally, algorithms have been assessed and shortlisted based on their problem similarity and features. The following algorithms have not been shortlisted:

- *DStPSO and GRASP* solve OP, which ignores the crucial Time Window element [5], [26].
- *Pareto Mimic algorithm, Ant Colony System and Adapted Genetic Algorithm* similarly ignore time windows while solving the potentially useful Time Dependency problem [17], [18], [30].
- *Branch and Price Algorithm, Tabu Search and Ant Colony System with Brain Storm Optimisation* all solve variants of VRP who's focus on minimising makespan is fundamentally different to the PIRP problem, efforts to change them would be significant. These algorithms will be revisited if particularly poor performance is experienced in the shortlisted algorithms [14], [22], [27].

The following algorithms were shortlisted for assessment:

- *Iterated Local Search (ILS)* demonstrated fast run-time, and is leveraged as a component of leading strategies [13], [15], [22] Fast run-time may be particularly relevant because of the likely use of a directions API.
- *Hybridized Simulated Annealing - Iterated Local Search (SAILS)* demonstrated high degree of accuracy, with additional computational overhead, which may limit the use of directions APIs for travel time.

- *Ant Colony Optimisation (ACO)* shown promise with a large number of constraints, in routing and scheduling. [10]
- *Adapted Genetic Algorithm* shows promise for a complete service algorithm which would not leverage a directions API due to its solution to Time Dependency. It provides a solution with moderate computational overhead and accuracy.
- *Ant Colony Optimisation* similarly shows promise for a complete service algorithm, again with moderate computational overhead and accuracy.

VI. Evaluation

As covered in the preliminary specification in Section I. The literature search assisted in clarifying the attributes of the PIRP problem, which have the following core criteria:

- *User preferences*: The value to the user in the suggested solution is near-optimal, compared with a manually selected optimal solution.
- *Alternative paths*: Presented to the user for selection.
- *Matches constraints*: The solution abides by the constraints set in the test cases.

The constraints and parameters are:

- *User modify nodes*: Users can add and remove properties if they meet the time criteria.
- *Time Windows*: Time windows are typical of real estate inspections (median of 30 mins), with set start-finish times.
- *Preferred Duration*: The user may set a duration they typically wish to spend at an inspection, these durations are respected.
- *Walking mode preference*: If within 10 minutes, walking will be preferred over other transport modes.

RQ1 will be addressed by a formal specification, extending Section II, with justification of each parameter presented. Detailed justification is a requirement to determine RQ1 answered. So far it has been determined that the minimal set of attributes should be chosen, as each additional criteria risks a significant computation time penalty [10]. An example of a rejected attribute is *scheduled meal breaks*, it was determined that this did not require formal scheduling.

Further, it was determined that multi-day routes will be considered in future work.

To address RQ2, a shortlist of models will be created. Their constraints and key variables will be listed and investigated whether minor modification can enable an existing journey planning algorithm. This will be determined by running the model and analysing which test cases above fail.

The model will likely be run in a public cloud environment *and* the configuration will be reported. The literature search did not reveal consistent reporting of environment configuration, therefore further analysis will be required to determine an environment that is typical, fair to the compared models (e.g. multi-core for parallel computing algorithm) *and* efficient at running the models under test. The same environment will be leveraged for model baselines.

Given this is a novel application, the popular approach of testing the model against standard benchmarks [16] will not be followed. The focus of this research will be to prove an algorithm can solve the Property Inspection Route Planning problem, not on algorithm run-time or accuracy. Future work will focus on potential run-time and accuracy improvements.

Web scraping will be used to source inspection data (location *and* start-finish times), these data points will be anonymised prior to use as data points. Data gathering is prone to bias, therefore a further literature search will be conducted to understand the main sampling considerations in the routing, scheduling *and* real-estate domains. (e.g., avoid duplicate listings *and* account for real vs. planned inspection start time variance).

To address RQ3, the existing list generation functionality run-time will be estimated by using web scraping, as leading MLS providers do not provide API access for public customer-facing functionality.

Given the potential variability of this approach, it will be conducted at least 10 times at different date-times, determined to be typical use hours (likely workday evenings and weekends) with average and standard deviation recorded.

A ranking approach will be used to create the property preference weightings. A best to worst scale described by Mueller et al. [23] will be used to infer weighted preference. This simplifies implementation, rather supplying a weighting for every property. The user will relatively rank their saved properties prior to list generation. Manual ranking may be replaced by recommendation engine rankings in future work. Several geographic samples will be taken with likely dominant transport modes paired with each sample, e.g. outer suburban and car, regional and car, urban and inner suburban and public transport.

VII. Expected Outcomes

It is anticipated the ideal algorithm will have faster run-time than the TRSP and (T)OPTW algorithms identified, due to narrower time windows and a simplified constraints set. Simplified versions of more recent Vehicle Routing algorithms may perform more optimally than OP algorithms, given the field seems slightly more mature, hosting several algorithms that have demonstrated success in parallel computation, despite the different approach to overall route length.

Network delay will become a prominent concern for this solution given the preference to utilise mapping APIs, therefore alternative approaches that minimise API calls will likely need to be explored. These approaches may include suburb level API calls or limiting pairwise API calls to properties that pass an approximated travel time test (based on geographic distance and assumed average speed).

Parking delay may require the adoption of assumptions related to availability based on collected council permit zone data. There is no central state or federal-level repository for Council permit zone data, so collection will likely be limited to the council areas in the test set. Future work may be spawned to collate a national repository of permit zones, potentially via image recognition approaches. Further, it is unlikely that a detailed approach to real-time availability will be employed due to the limited footprint of freely available sensor data. Of the regions explored, only Melbourne City Council offers live and free parking sensor data, see [8]. If adopted by more councils an approach as described in [33], may be employed, where open parking sensor data is leveraged to predict availability in Melbourne and San Francisco.

REFERENCES

[1] R. A. Abbaspour and F. Samadzadegan. Time-dependent personal tour planning and scheduling in metropolises. *Expert Systems with Applications*, 38(10):12439–12452, 2011.

[2] F. Arnold, M. Gendreau, and K. Sörensen. Efficiently solving very large-scale routing problems. *Computers & Operations Research*, 107:32–42, 2019.

[3] Australian Bureau of of Statistics. Media Release - Census: younger Australians more likely to make a move (Media Releases), 2017, Oct 2017.

[4] R. Bellman. Dynamic programming treatment of the travelling salesman problem. *Journal of the ACM (JACM)*, 9(1):61–63, 1962.

[5] V. Campos, R. Martí, J. Sánchez-Oro, and A. Duarte. Grasp with path relinking for the orienteering problem. *Journal of the Operational Research Society*, 65(12):1800–1813, 2014.

[6] I.-M. Chao, B. L. Golden, and E. A. Wasil. The team orienteering problem. *European journal of operational research*, 88(3):464–474, 1996.

[7] A. J. Christianto, P. Chen, O. Walawedura, A. Vuong, J. Feng, D. Wang, M. Spichkova, and M. Simic. Enhancing the user experience with vertical transportation solutions. *Procedia computer science*, 126:2075–2084, 2018.

[8] M. C. Council. On-street parking bay sensors: Open data: Socrata, Apr 2020.

[9] M. Dawson, D. Burrell, E. Rahim, and S. Brewster. Integrating software assurance into the software development life cycle (sdlc). *Journal of Information Systems Technology and Planning*, 3:49–53, 01 2010.

[10] W. Deng, J. Xu, and H. Zhao. An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem. *IEEE access*, 7:20281–20292, 2019.

[11] D. Feillet, P. Dejax, and M. Gendreau. Traveling salesman problems with profits. *Transportation science*, 39(2):188–205, 2005.

[12] D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou. Mobile recommender systems in tourism. *Journal of network and computer applications*, 39:319–333, 2014.

[13] D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou. A survey on algorithmic approaches for solving tourist trip design problems. *Journal of Heuristics*, 20(3):291–328, 2014.

[14] M. Gmira, M. Gendreau, A. Lodi, and J.-Y. Potvin. Tabu search for the time-dependent vehicle routing problem with time windows on a road network. *European Journal of Operational Research*, 288(1):129–140, 2021.

[15] A. Gunawan, H. C. Lau, and P. Vansteenwegen. Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, 255(2):315–332, 2016.

[16] A. Gunawan, H. C. Lau, P. Vansteenwegen, and K. Lu. Well-tuned algorithms for the team orienteering problem with time windows. *Journal of the Operational Research Society*, 68(8):861–876, 2017.

[17] L. Ke, L. Zhai, J. Li, and F. T. Chan. Pareto mimic algorithm: An approach to the team orienteering problem. *Omega*, 61:155–166, 2016.

[18] J. Li. Research on team orienteering problem with dynamic travel times. *JSW*, 7(2):249–255, 2012.

[19] K. H. Lim, J. Chan, C. Leckie, and S. Karunasekera. 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.

[20] H. R. Lourenço, O. C. Martin, and T. Stützle. Iterated local search. In *Handbook of metaheuristics*, pages 320–353. Springer, 2003.

[21] S. Mathew and J. Joseph. Decision fatigue. *LISTENING TO CONSUMERS OF EMERGING MARKETS*, page 175, 2014.

[22] I. Mathlouthi, M. Gendreau, and J.-Y. Potvin. Branch-and-price for a multi-attribute technician routing and scheduling problem. In *SN Operations Research Forum*, volume 2, pages 1–35. Springer, 2021.

[23] S. Mueller, I. Francis, and L. Lockshin. Comparison of best–worst and hedonic scaling for the measurement of consumer wine preferences. *Australian Journal of Grape and Wine Research*, 15(3):205–215, 2009.

[24] J. Nielsen, Aug 2001.

[25] M. P. Peterson. Evaluating mapping apis. In *Modern Trends in Cartography*, pages 183–197. Springer, 2015.

[26] Z. Sevkli and F. E. Sevilgen. Discrete particle swarm optimization for the orienteering problem. In *IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2010.

[27] Y. Shen, M. Liu, J. Yang, Y. Shi, and M. Middendorf. A hybrid swarm intelligence algorithm for vehicle routing problem with time windows. *Ieee Access*, 8:93882–93893, 2020.

[28] T. Tsiligirides. Heuristic methods applied to orienteering. *Journal of the Operational Research Society*, 35(9):797–809, 1984.

[29] F. Ullah, S. Sepasgozar, and S. Siddiqui. An investigation of real estate technology utilization in technologically advanced marketplace. In *9th International Civil Engineering Congress (ICEC-2017),"Striving Towards Resilient Built Environment", December*, pages 22–23, 2017.

[30] C. Verbeeck, K. Sörensen, E.-H. Aghezzaf, and P. Vansteenwegen. A fast solution method for the time-dependent orienteering problem. *European Journal of Operational Research*, 236(2):419–432, 2014.

[31] C. Verbeeck, P. Vansteenwegen, and E.-H. Aghezzaf. The time-dependent orienteering problem with time windows: a fast ant colony system. *Annals of Operations Research*, 254(1):481–505, 2017.

[32] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann. Time-aware point-of-interest recommendation. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 363–372, 2013.

[33] Y. Zheng, S. Rajasegarar, and C. Leckie. Parking availability prediction for sensor-enabled car parks in smart cities. In *2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, pages 1–6. IEEE, 2015.

[34] L. Zhu and J. D. Gonder. A driving cycle detection approach using map service api. *Transportation Research Part C: Emerging Technologies*, 92:349–363, 2018.