A Distributed Approach to Constructing Travel Solutions by Exploiting Web Resources
Oudom Kem, Flavien Balbo, Antoine Zimmermann

To cite this version:

HAL Id: emse-01412921
https://hal-emse.ccsd.cnrs.fr/emse-01412921
Submitted on 3 Jan 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
A Distributed Approach to Constructing Travel Solutions by Exploiting Web Resources

Oudom Kem, Flavien Balbo and Antoine Zimmermann
Univ Lyon, MINES Saint-Étienne, CNRS, Laboratoire Hubert Curien, UMR 5516, F-42023, Saint-Étienne, France
Email: {oudom.kem, flavien.balbo, zimmermann}@emse.fr

Abstract—Many advanced traveler information systems (ATIS) provide travel solutions that are limited, by technical obstructions or by design, in terms of geographical coverage, transport services, and/or travel modes. However, using existing ATIS in an integrated manner can broaden the coverage of travel solutions, while preserving the advantages of each system. This paper presents an approach to exploit web resources such as ATIS, data sources and services to construct travel solutions. More precisely, it focuses on itinerary formulation and discovery of resources relevant to each itinerary. We propose a semantic model for describing and interlinking geographic entities in relation to web resources, creating a graph of related entities. Itinerary formulation and resource discovery are achieved via exploring the graph. To this end, we propose an adapted version of multi-agent A* algorithm.

Keywords—Multi-agent A* algorithm; Seed graph; Semantic Web; Travel solution

I. INTRODUCTION

Numerous advanced traveler information systems (ATIS) are in operation. However, most ATIS are designed to cover only certain types of travel modes, transport services, and/or geographical areas. For instance, there are systems designed only for private vehicles such as PersianGulf [1], for public public transport (e.g. TCL for public transport in Lyon) and for a specific geographical coverage (e.g. Transport for London for the city of London). Therefore, regardless of the significant number of existing ATIS, travelers are often required to consult multiple ATIS and/or to seek for travel-related information from various other sources to acquire sufficient information for their trip. When unexpected events such as delays or accidents occur during their trips, travelers need to search for alternatives to adapt to those events.

Pertinent resources for building travel solutions, such as ATIS and other data sources as well as services are available on the Web. Using them complementarily can improve three main aspects of travel solutions. (1) Adaptability: Solutions can be built using resources, which are not statically predefined, but are chosen according to actual requests from travelers. (2) Completeness: The use of multiple existing ATIS as well as other resources in an aggregated manner can widen the coverage of solutions. (3) Personalization: Travelers’ preferences and constraints can be taken into account both in the process of discovering resources and in the computation of solutions.

The aim of our research is to design a framework that is able to automatize the entire process of formulating itineraries, searching for relevant web resources, accessing them and integrating acquired information and services to construct travel solutions tailored to each traveler’s request. Fig. 1 illustrates the abstract model of the framework, emphasizing the four integral components. Request analysis component extracts essential information such as the origin, the destination and any indicated preferences or criteria from a given request. Discovery component searches for an optimal itinerary between an origin and a destination as well as resources that provide information and/or services assisting travel along an itinerary. Taking an itinerary, its relevant resources, and travel information as well as services acquired previously as an input, solution construction component’s roles comprise integrating the acquired heterogeneous data, coordinating services, retrieving travel-related data and services (e.g. weather, gas stations) if necessary, and eventually assigning information and services to relevant segment(s) of the itinerary to build a travel solution. Solution refinement component monitors resources associated with a chosen itinerary for updates and unexpected events (e.g. delays, accidents), and adapts solutions to consider changes.

The focus of this paper is the discovery component. A distributed approach to itinerary formulation and web resource discovery is presented. In this approach, a semantic model for describing and interlinking geographic/administrative entities in function of web resources is proposed. We adapt a multi-agent A* algorithm to perform resource discovery and itinerary formulation.

II. SEED GRAPH

An itinerary is composed of one or more segments. Each segment consists of a starting and an ending locations that are geographic entities. Generally, geographic entities can be under a certain hierarchy, namely administrative entities (e.g. a city, a country). We use the notion of seed as an intermediary between geographic/administrative entities and resources relevant to the entities. Each seed describes an entity in terms of resources relevant to that entity and connections to other entities in function of connecting resources.

These connections link each seed to other related seeds, and thus forming a seed graph. Seed data model consists of:
Continuous refinement and adaptation of the solution
Request
graph. Let us examine one seed in the graph. The Saint-
´Etienne topic (i.e. topic of the target seed). SNCF, AirFrance and Megabus
provide information on how to travel from Saint-
´Etienne to Paris.

Therefore, ideally, from people involved in the domain (e.g.
authority of a city, transport operators) to individuals willing
to contribute, anyone is able to create seeds for a set of
geographic and/or administrative entities of their interest,
link them to existing seeds, and make them available on the
Web. In the same manner as how the Web has evolved so
far, this would allow the seed graph to scale and grow.

III. ALGORITHM FOR ITINERARY FORMULATION AND RESOURCE DISCOVERY

A. Problem description

Given a seed graph, itinerary formulation and resource
discovery are realized by exploring the graph. This explo-
ration can be formulated as a search problem as follows:

• States: A set of states \( S \) corresponds to seeds in a graph.
• Initial state: An initial state \( s_i \in S \) corresponds to the
seed whose topic is the origin of a requested itinerary.
• Actions: Actions are connecting resources of connec-
tions among seeds. In each state, there is a set of
applicable actions, referring to connecting resources of
connections to other seeds that can be found in a seed.
• Transition model: Executing an action \( a \in Actions \)
in a state \( s \in S \) leads to a resulting state \( s' \) which
is the target seed of a connection. In our problem,
actions and their cost are initially unknown, and are
to be discovered during the search.
• Goal test: A state \( s \) is a goal state if its topic is the
destination of a requested itinerary.
• Path cost: Path cost is the actual travel cost such as
travel distance, duration, and monetary cost.

Solution consists of a path from an initial state to a goal
state (i.e. itinerary) and a sequence of actions (i.e. connecting
resources providing instructions on how to travel along the
itinerary) leading to the goal state. Referring to the graph in
Fig. 2, if the request is to travel from Saint-
´Etienne to New
York, the problem is to find a path between Saint-
´Etienne and New York and its connecting resources. Let us suppose
that the best solution is:

\[ \text{Saint-
´Etienne} \xrightarrow{SNCF} \text{Paris} \xrightarrow{AirFrance} \text{New York} \]

The itinerary is to travel from Saint-
´Etienne to Paris,
and then to New York. Being web-based ATIS, SNCF and
AirFrance provide information on how to travel from Saint-
´Etienne to Paris and from Paris to New York, respectively.
B. The discovery algorithm

The proposed algorithm is based on the classic A* algorithm and multi-agent approach. Some basic parts concerning the usage of agents (e.g. exchanging messages) are inspired, with adaptations to our problem, from a generic multi-agent version of A*, MA-A* [4]. Multi-agent systems have been shown to be suitable for solving problems in an open and complex environment where required resources are distributed [5]. They are applicable in our context because seed graph and resources are distributed over a highly dynamic and open environment, the Web. There are some distributed and parallel variants of A* such as [4] and [6]. While many works address the issues of distributed and parallel settings such as separating search spaces and distributing knowledge, none, to our knowledge, address the dynamic and open environment, the Web. There are some open and complex environment where required resources are inspired, with adaptations to our problem, from a generic algorithm and multi-agent approach. Some basic parts concerning the usage of agents (e.g. exchanging messages) are inspired, with adaptations to our problem, from a generic multi-agent version of A*, MA-A* [4]. Multi-agent systems have been shown to be suitable for solving problems in an open and complex environment where required resources are distributed [5]. They are applicable in our context because seed graph and resources are distributed over a highly dynamic and open environment, the Web. There are some distributed and parallel variants of A* such as [4] and [6]. While many works address the issues of distributed and parallel settings such as separating search spaces and distributing knowledge, none, to our knowledge, address the dynamic and open environment, the Web. There are some open and complex environment where required resources are inspired, with adaptations to our problem, from a generic algorithm and multi-agent approach. Some basic parts concerning the usage of agents (e.g. exchanging messages) are inspired, with adaptations to our problem, from a generic multi-agent version of A*, MA-A* [4]. Multi-agent systems have been shown to be suitable for solving problems in an open and complex environment where required resources are distributed [5]. They are applicable in our context because seed graph and resources are distributed over a highly dynamic and open environment, the Web. There are some distributed and parallel variants of A* such as [4] and [6]. While many works address the issues of distributed and parallel settings such as separating search spaces and distributing knowledge, none, to our knowledge, address the dynamic and open environment, the Web. There are some open and complex environment where required resources are

Algorithm 1 Discovery algorithm for a search agent

```
1: while no verified solution found do
2:   for all messages m received do
3:     process-message(m)
4:     s ← pop-min(OL)
5:     expand(s)
6:     a ← pop(AL)
7:     request-action-cost(a)
8:     add a to PL
9:     broadcast action-occupation-message(a)
```

Algorithm 2 expand(s)

```
1: if s is a goal state then
2:   broadcast goal-verification-message(s, f(s), ID)
3:   if verify-goal(s) returns false then
4:     put s back into OL
5:     return
6:   broadcast expansion-message(s, f(s), ID)
7:   add s to CL
8:   actions ← get-applicable-actions(s)
9:   add actions to AL
10: broadcast new-actions-message(actions)
```

An action-occupation-message about a is broadcasted, and receiving agents update their AL accordingly. Executing a in s will lead to a successor s'. To calculate the cost of s', f(s'), it is necessary to acquire the cost of a, c(s, a, s'). A sends a non-blocking request for c(s, a, s') to an RA associated with a. Once it receives the response in the form of response-action-cost message, it computes f(s'), s' is added to OL if it is not in OL and CL. If s' is in the OL, but its cost is higher than the new s', it is replaced by the new s'. In case s' has already been expanded (i.e. in CL) and the new s' has a lower cost, s' is re-opened. Otherwise, it is discarded.

1) Goal verification procedure: When a goal state is expanded, the expanding agent broadcasts a goal-verification-message to verify the optimality of the solution. A solution with a goal state s* is optimal if there exists no state s in the entire system where f(s) < f(s*). To verify that property, assuming all messages in the queue are processed, each agent checks its OL, AL and PL. If there is any candidate state s in OL where f(s) < f(s*), the solution is not verified. If there are actions in AL or PL, the solution is not verified. The cost of the actions in these two lists are still unknown, so it is impossible to determine the cost of their resulting states. If the solution is not verified, s* is reinserted into OL. In this way, only states having lower cost than s* will be expanded. If a better goal state is discovered, it replaces the existing goal state in OL. When a solution is verified, the expanding agent traces back for the path and executed actions. Then, it broadcasts a terminating message.

2) Concurrent action selection: If multiple agents have taken the same action, the first agent, determined by the time-stamp included in the action-occupation-message, that chose the action can execute the action. If the time-stamps
detect a concurrent expansion of the same state. To prove the optimality of our algorithm, we employ the proof by contradiction by assuming the contrary. We consider 3 following cases as also addressed in the generic MA-A* [4]. (1) The algorithm terminates at a non-goal state. This case is impossible because the goal verification procedure only starts when expanding a goal state. (2) The algorithm terminates at a non-optimal goal state. In the goal verification procedure, a goal state \( s^* \) is verified only when, for every agent, it has the lowest cost in the OL, and there exists no action in the AL or PL that may lead to a state \( s \) where \( f(s) < f(s^*) \). Therefore, when a non-optimal goal is being verified, there must be a state \( s' \) belonging to the optimal path, in the OL or as a resulting state of an action in the AL or PL, where \( f(s') < f(s^*) \). In such case, \( s^* \) is not verified. (3) The algorithm does not terminate. First, if an optimal solution exists, the algorithm terminates after the solution is verified. Second, when all states in the search space are expanded, the algorithm terminates. Third, the search space is finite, so the number of time a state \( s \) can be reopened from the CL is also finite.

IV. Conclusion

The proposed approach allows independent resources to be described and interlinked such that they can be discovered. The distributed and open architecture enables flexible entries and exits of resources. Moreover, employing multi-agent approach facilitates the use of distributed and heterogeneous resources and the personalization process. The feasibility of the approach depends on people’s will to describe and interlink seeds by adhering to a common model, but a worldwide adoption of a specification is not uncommon on the Internet. However, the number of web resources are immense. The scalability of the approach needs to be ensured and will be addressed in our future work.

REFERENCES