Extending Query Fulfillment by Digital Personal Assistants with Temporal Reasoning

Andrew Almaguer

Institute of Architecture of Application Systems

Abstract. Digital Personal Assistants like the offerings from major tech companies like Amazon, Apple, and Google are seeing increased adoption rates as they near the end of their first decade of existence. Despite the backing of these companies, theses assistants remain restricted in the types of queries that they can provide answer to. Instead of being an intelligent assistant to users, the current trend shows a move to a new way to search and a focus of very specific experiences. This paper considers an alternative to how queries can be understood, temporal reasoning. With this type of enhancement, the possibilities for understanding and fulfilling user requests would become more various. This work thus seeks to investigate the application of Allen's Interval Algebra on a potential user query. This is shown to provide some benefit and opens questions to its potential alternative applications and usage.

Keywords: digital personal assistants, temporal reasoning, query fulfillment

1 Introduction

Digital personal assistants are tools that have managed to reach maturity and fairly wide adoption since their introduction in the early 2010s. Beginning with the introduction of Siri in October 2011 and Google Now in July 2012 and coinciding with the smartphone era, digital personal assistants have become an ever present and useful tool in the daily lives of many people. Although these assistants have continued to improve, one could argue that we have begun to reach a level where assistants have little room left to become better at performing the types of tasks they are currently capable of. This can be seen simply from the fact that the three most popular digital assistants share mostly the same features. It may be time that these assistants are improved by starting to work on forming a more complex understanding of the elements that make up the queries they are often presented.

1.1 Digital Personal Assistants

As already mentioned, Apple's Siri has been around since October 2011 [14]. Google's Google Now appeared the following year [11] and their Google Assistant, which could be looked at as an evolution of Google Now, appeared in 2016 [8]. Amazon released Alexa in 2014 [6].

While there are other assistants like Microsoft's Cortana and Samsung's Bixby; Siri, Google Assistant, and Alexa account for the overwhelming majority of actual consumer use according to a 2019 Microsoft report [19]. This is attributed in the report to the fact that Google Assistant and Siri are widely used on smartphones and Alexa in smart speakers, which are markets that Microsoft, for example, do not have a direct share of. Our work will mostly focus on these three digital assistants.

Base Features If you take a look at the promotional material for these three major assistants, you may notice how many of the touted features are the same across the board. Most understood commands come down to either a simple command or a simple research-able question. Commands can be boiled down to a simple verb like 'find' or 'play' and a few words to provide more context to fulfill the command. Questions, on the other hand, tend mostly towards simple research queries like "What time does the supermarket close?" or "What will the weather be like tomorrow?". These are akin to what one might quickly look up with a search engine search or check on a daily basis. One further category that all seem to have is a small selection of specialized queries like "Tell me a joke". A deeper analysis of the queries and functions promoted for each of these assistants will be done in the course of this work.

Extensions In addition to the base features, all three also offer some kind of API for developers. SiriKit [16], Actions on Google [3], and the Alexa Skills Kit [7] provide a way to receive the information that is gleaned during the processing of the voice query. This is limited as it does not extend the base ability of that specific assistant, rather it provides a way to get some information following some processing by the assistant to your code where you then may be able to do something more complex.

In many of the advertised use cases this results in providing similar features to those already covered in the base features, but executed through your code because the user specifically mentioned your app. SiriKit is a good example of this exact issue as all examples provided on their website fall into the first category of queries mentioned in the previous paragraph (simple commands) with the addition of your application's name [16]. This type of extremely limited approach is stifling to developers wishing to provide their users with complex services.

Both Amazon and Google also provide some case studies to show how developers take advantage of these development kits. There does not seem to be a big difference between each case study so we will just look at one from each source. Google has an example case that showcases how to create something a bit more interactive than simple commands or questions, a game. The Mad Libs Action [13] allows users to play a short game where they are prompted to provide words to finish a story. There is nothing here that goes outside of the normal base abilities of Google Assistant as the conversation-like style of interaction used here is also supported by many assistants [3].

Headspace is an app that seeks to help users with meditation. The Alexa case study on Headspace [12] does not mention anything groundbreaking either, rather more of a specialized and directed experience. This is highlighted by "The Alexa experience features custom audio to lead guided exercises, as well as a robust conversational repertoire with nearly 500 responses and reprompts." While seemingly impressive, this shows that there is a very specific list of interactions that are supported with no ability to have any kind of dynamic experience. It seems from all examples that even the development kits that are offered are limited by the base abilities of each assistant.

The Viv Presentation One could certainly look at all of the major assistants and come to the conclusion that they are all at a similar stage because that is just what our level of technology is capable of. However, there is a small example of a case that showed this may not be true, the 2016 Disrupt NY conference presentation on Viv [2] showcased an assistant prototype that had the ability to dynamically generate a program to fulfill the understood goals of the user.

The presentation, although short and very light on technical details, showed promise in fulfilling more complex queries through service composition. The presenter claims base features of current assistants are like that because there will be a mostly fixed program in the background that gets called with a bit of the information provided in the query. Viv seemingly had the ability to do more processing on each part of the query in order to understand much more complex queries. While this will not be the direct goal of our work here, it provides a good example of how we may need to start thinking to make digital personal assistants and similar systems more capable.

1.2 Improving Reasoning

After this short look into the current state of digital assistants, it is clear that we can do more to improve them. These assistants are complex systems that are capable of doing amazing things like understanding user intent, but no one would confuse them with a human with a more complex understanding of all elements of a query. For the most part, they see elements as fixed and lack the reasoning to work past that assumption. Maybe this type of reasoning is ultimately not necessary, but that is something we have yet to really delve into. That is the goal of this work, to see if a small improvement in reasoning about a single element in a query can provide some benefit.

1.3 Outline of This Work

The remainder of the paper starts with a deeper look at the current state of digital personal assistants as well as a look into some related work (Sect. 2). Following that we dive into the design of our experiment (Sect. 3) and look at the results (Sect. 4). We will then continue with a short critical look at our experiment (Sect. 5). Finally, we provide an overall conclusion and a short outlook on possible future work (Sect. 6).

2 Research and Current Trends

The primary focus on this work is to look into ways to improve the reasoning ability of digital personal assistants. Unfortunately, there does not seem to have been much research into this field. So instead of this section being a review of current literature it will be a look into the state and trends in the world of digital personal assistants. Although we cannot look directly into the inner workings of the major players in this field, we do have quite a bit of research from them giving us an idea of the current state of digital assistants.

2.1 Microsoft's Voice Report

In a Microsoft published report from 2019 [19], we are given a very good overview of which assistants are popular, how much they are used, and for which jobs users currently task them with. The report has four main sections: usage, trust, functionality, and shopping. We will not be going into the trust and shopping sections, which focus on security and marketing, as they are not relevant for this work.

Usage The usage section reports the results of two surveys that were conducted on digital personal assistants. Along with confirming the expectations one would have about which assistants are most popular and why, the results presented in the report paint a good picture of the current state of those assistants. Some of the relevant statistics:

- Siri and Google Assistant are the most popular, with Alexa (and Cortana) slightly behind
- Popularity is attributed partially to overall smartphone adoption, in the case of Siri and Google Assistant, and rising smart speaker adoption in the case of Alexa
- Approximately 70% have used assistants and 50% extensions like Alexa skills or Google actions
- While the majority of users are using smartphones or smart speakers, IoT applications like smart TVs or smart autos are also used significantly
- 80% reported being satisfied with their personal assistant interactions
- 66% reported using digital assistants weekly and 19% daily
- Smart speakers are used to control smart homes with controlling music and lighting being the main applications. Controlling the security system and thermostat were also reported as significant use cases.
- The most popular tasks are simple ones (participants were asked to select five tasks), the top 5 answers and summary of the rest

- Searching for a quick fact (68%)
- Asking for directions (65%)
- Searching for a product or service (52%)
- Searching for a business (47%)
- Searching for a product or service (44%)
- Tasks related to shopping made up the rest of the more used answers, with most falling between 10% and 25%. Only 5% answered "other".

Some important conclusions can be drawn from these results. One conclusions made in the report is that based on "Diffusion of Innovation" theory, digital personal assistants are "on the cusp of going mainstream." This was specifically linked to the fact that 19% reported using their assistants daily which can be interpreted as a sign that assistants are now leaving the early adopter phase. One other important conclusion can be made about what I claimed in the introduction, assistants are limited in what they can do. The most commonly performed tasks are all very simple queries, many being direct replacements for simple web searches. This hints at the assumption that assistants lack the ability to be truly useful in new ways and are often instead a replacement input method for tasks that were commonly done in the past.

Functionality The second most important section of this report for the purposes of this work is on functionality. The authors assess the focus of the major players in four categories: Knowledge, Productivity (and Work), Utility (IoT and Home Management), and Commerce. Unsurprisingly given the developer, Alexa was rated as hiving a higher focus on Commerce and in Utility. Siri was found to have their focus more in the directions of Productivity and Knowledge. Google Assistant was overall more well rounded than either of its closest competitors, but with slightly more focus on Commerce than anything else in particular.

Cortana was also rated here, which is expected given it is a report from Microsoft. One interesting thing that is being done differently with Cortana is that Microsoft plans to release their assistant as a dedicated action and skill for Google Assistant and Alexa. This would allow them to tap into the existing user bases, but may also be limiting as they have less control over the base abilities of the assistants in this case.

2.2 Advertising Analysis

Building on the results of this report, I have done a more in-depth look at and analysis of the core functionalities of each of the major assistants. I made one major assumption here, the marketing for each assistant is the best representation of what it is capable of. All three development companies are competing with each other to win users, so this was a reasonable assumption from my point of view. With that in mind, I created lists of the commands that were advertised and analyzed them. I categorized them and also did some comparisons between them all. See the appendix for a more detailed list of the commands I found and used in this analysis.

Table 1. Simple Comparison

	Alexa [4]	Google Assistant [9]	Siri [15]
Total # of Commands	68	48	32
Tasks	36 (52.9%)	28 (58.3%)	20 (62.5%)
Queries	20 (29.4%)	14 (33.3%)	10 (31.3%)
Other	12 (17.6%)	6 (14.3%)	2 (6.3%)

Command Types Based on the simple comparison in Table 1, there is a clear order of importance of the basic types of commands. Tasks, the category in which I placed commands that fell into the form "verb + context information" like "Call Mom" or "Tell me the latest news", are the obvious top focus. This is not a surprise as one of the draws of digital assistants is to get help performing common tasks. Queries, which compromise of commands in the form "question word + context information" like "What is the weather like today?" or "How do you say 'thank you' in Mandarin?", were the second most common example. This can be seen as a new interface for search engines, although some of the examples also included personalized use cases like "What is the temperature in here?" or "What is on my schedule today?". The other category is where I placed commands that are specialized for that particular assistant, like "Tell me a joke" or "What is your favorite number?". These commands are often more of a novelty than anything. This could be an explanation for why Siri does not advertise them, as many might see these commands at novelty and not as a reason to explore using a particular digital personal assistant.

Command Categories To finish this short in-depth look into how digital personal assistants are advertised, it would be beneficial to look at the actual queries in a bit more depth. In the list of commands (Alexa A, Google Assistant B, Siri C) included in the rough analysis we just concluded, I also made a simple categorization of the command. The categories I decided upon were Entertainment, Productivity, Communication, Knowledge, Smart Home/IoT, Shopping, and Specialty. It is certainly possible that commands actually fall within two categories, but I limited each command to a single category.

Alexa commands covered the widest range of categories which may not be surprising since the web pages listing the features of Alexa are also much more thorough than their counterparts. Alexa also was the only one to feature commands related to shopping. While it is not surprising that Amazon uses their assistant to drive more sales in their store, it is a bit surprising that neither Google nor Apple focus on shopping.

Another category that Google ignores in their marketing is Smart Home/IoT. Google's Nest [10] devices are Smart Home devices like thermostats, cameras, and locks, which certainly work with their digital assistant. Given that Google themselves produce such devices shows that it is clearly a focus, which makes this absence in their advertising noteworthy.

Siri has which could be considered the most complex commands, a few of which seem to do more analysis of the person using the assistant. “Open the presentation I was working on yesterday” or “What did they say?” (used while streaming a video) are meant to accomplish something more complex than just doing exactly what the user said. These queries require more context and show a much larger focus on an integration between Apple products and services. Overall, many of the commands are ultimately very similar and it’s obvious that each one of these assistants can accomplish the types of commands that were found to be most used in the previously discussed Microsoft report.

2.3 Digital Assistants in Marketing

To round out our look at the current state we will be looking at an article from Matt Mierzejewski that appeared on Google’s Think with Google website. This is a website that is more dedicated to marketing than anything else and thus often provides statistics and analysis on current trends. The article, titled “Digital assistants are the next marketing revolution. Here’s how to be ready” [18] makes an argument that is contrary to what this work is proposing, but more in line with what Amazon, Apple, and Google are focusing on.

Mierzejewski strongly suggests that companies that do their best to understand how users interact with their services and are able to successfully tailor the user experience will be in the best position for the rise of digital personal assistants. This clearly falls in line with what is possible for developers using Actions, Skills, and SiriKit. The suggestion made here would lead to a situation where each company or brand tailors a very specific experience. Although that may lead to digital assistants seeming more natural during that very specific interaction, it will do little to expand the natural abilities of the assistants. I do not feel that Mierzejewski’s suggestion needs to exist separately from what our work is proposing, but it does shed some light on the current focus on the major players in the consumer market.

3 Experiment Design

The current major digital assistants have a set of features that is for the most part uniform and, when broken down to their basic elements, fairly simple. While assistants often also have the ability to remember “conversations” or have their features expanded by developers, the SDKs available limited these abilities to working in similar ways to main abilities that the assistant has. A possible way to address these limitations is to work on ways to help digital personal assistants to have a more complex understanding of some of the important individual elements. This is the goal of our experiment.

3.1 Assumptions

In order to keep the experiment focused, I wanted to make a few assumptions about the context. During the preliminary discussions on this work, it was decided to focus on a single query, in which all elements except for one would remain fixed. Ideally, the variable element that we would be testing would also be something that current assistants do not have a very complex understanding of. However, this specific goal turns out to not be particularly helpful as assistants mostly have a very basic understanding of the possible variables that were considered.

In addition to keeping a fixed query outside of a single variable, I also wanted to remove one of the other major features of digital personal assistants as a possible factor, language processing. Addressing the ability to understand more complex queries is not in line with the general goal. What is in line with that goal is finding a way to give assistants a more complex understanding of simple elements. Because of this, I also made the assumption that queries are simply understood and that is why we will only focus on improving a specific stage of processing that query.

3.2 Variable Selection

With the discussion of assumptions made in mind, I looked a which variable could be improved with more reasoning ability. If you consider the average task-style query, they are almost always of the form "imperative verb + context information". It is this context information that contains the variables that were considered. While digital assistants surely receive some more complex information about individual elements in this context, these elements are often not treated or processed in any special way. If these elements happen to be temporal or spatial, rather than just standard factual elements, they provide an interesting opportunity, as one can use fairly complex reasoning when using them. More specifically, spatial and temporal elements can be understood as a singular, complete object or a combination of various smaller parts.

In the case of some spatial elements, it is very clear how they can be broken down into parts as they are already done so in some cases. Cities often have districts or neighborhoods, for example. If a user had a query related to a specific city, then it could be useful to use this information as well. This is already done in many cases, like in real estate or tourism.

This does not only apply to cities, states, or countries. Another example could be office buildings or apartments. Digital personal assistants could, with a better understanding of the space they operate in and spatial reasoning, improve their ability in Smart Home/Office and Productivity fields. It is certainly imaginable that some day offices will become more energy efficient through better understanding and use of their space, a change that could be helped along by assistants.

Temporal elements are even more conveniently broken down into elements like minutes, hours, and days. The interesting thing about these in comparison to spatial elements is that they have a "fixed" value, an hour is an hour as opposed to one city neighborhood being equal to another. The fact that these values also directly correspond to others and have clear meaning for us as the user of a service is also very helpful. A day can be split up into various combinations of hours and an hour can be split up in the same way with minutes.

This attribute of temporal elements provides a possibly very useful way to add something "dynamic" to how temporal elements are understood by digital personal assistants. When given a start and end date or time, should that duration be understood as a singular fixed value or could it be useful to view it as a combination of a number of start and end dates or times? This question is one of the primary questions that led us to looking deeper into how to provide a way to integrate better temporal reasoning.

3.3 Allen's Interval Algebra

The "modularity" of temporal values gives rise to an interesting opportunity to take a logical approach to working with these values. One of these approaches is Allen's Interval Algebra. Proposed by James F. Allen in 1983 [17], Allen's Interval Algebra defines 13 relations between a pair of time intervals. The list of relations can be seen in Table 2. Of these 13 relations, 6 are actually inverses of other relations, meaning that inverting the order of intervals when using an inverse produces equivalent statements (for example, X $<$ Y and Y $>$ X are equivalent).

Table 2. Interval Relations [5]

Relation	Interpretation
X $<$ Y	X takes place before Y
X $>$ Y	X takes place after Y
X m Y	X meets Y
X mi Y	X (inversely) meets Y
X o Y	X overlaps with Y
X oi Y	X (inversely) overlaps with Y
X s Y	X starts Y
X si Y	X is started by Y
X d Y	X is during Y
X di Y	X encompasses Y entirely
X f Y	X finishes Y
X fi Y	X finishes in tandem with Y
X = Y	X is equal to Y

Even with the use of just a small set of these relations, Allen's Interval Algebra provides an interesting opportunity to accept a given fixed interval and do more with it by splitting it up. This idea is what this work seeks to implement and analyze.

3.4 Algorithm Design

With the preliminary discussion on some of our assumptions and an introduction to an integral part of this work, we can now begin looking deeper into exactly what will be implemented as part of this work. The idea for this work began with a question: what if you took an event that takes place over a sizable interval and instead split it up into smaller intervals of that event?

The specific example that has decided upon is taking a request to book a hotel. This request would normal have its start and end dates used to find hotels that are available for the whole interval, which the user can then book. However if you combine with with our initial question, another question arises: what if it were acceptable to book a collection of hotels that cover that whole interval? If a specific hotel is unavailable for a specific date during a given interval, in normal circumstances it would normally be removed from the choices. One could instead query the possible sub-intervals and then combine the best choices to find a solution for the entire interval.

In order to accomplish this, there are a clear number of steps. The general implementation will be detailed here. The steps I was able to deduce are as follows:

1. Given a start and end date, create a matrix of all possible sub-intervals
2. Use this matrix to send requests, append this information to the matrix
3. Use the now saved information to determine the value of the interval based on a desired value function
4. Use the values to find an optimal solution that exactly covers the initial input interval

I will be going further into the first, third, and fourth of these steps. The second step is trivial in terms of design, but could be complicated in terms of implementation. It depends on the source being used and since this discussion is about the general algorithm, it has no place here. The third and fourth steps do not necessarily need to be done in this order and they can even possibly be combined. For the purposes of this work, I will not go into much depth into this value determination step, but it will appear a few times during our discussion.

Interval Matrix We begin with the creation of a matrix of intervals (see List. 1). Essentially, we begin with the length of the intervals we want to create. We then iterate through the initial user-provided interval to create all possible sub-intervals of this length. All sub-intervals are appended on to a list. These lists are all appended to a list “of lists”, resulting in what is essentially a 2D Matrix.

```
INPUT:
  start date Start_Date
  end date End_Date
OUTPUT:
  matrix of sub-intervals in Interval(Start_Date, End_Date)

BEGIN
  NEW Interval Base_Interval = Interval(Start_Date, End_Date)
```

```
  NEW List[List[Interval]] 2DMatrix_of_Intervals = [[]]

  # Populate 2DMatrix_of_Intervals
  FOR Integer i IN Range(1, Base_Interval.duration.days):
    # i is the duration of the Intervals that will be created
    NEW List[Interval] List_of_Intervals = []

    FOR Integer d IN Range(0, Base_Interval.duration.days - i):
      # d is the offset that will be added to the start of
      # the created Interval
      NEW Date New_Start_Date = Start_Date + OffsetDays(d)
      NEW Interval New_Interval =
        Interval(New_Start_Date, New_Start_Date + OffsetDays(i)

      List_of_Intervals.append(New_Interval)
    ENDFOR

    2DMatrix_of_Intervals.append(List_of_Intervals)
  ENDFOR

  RETURN 2DMatrix_of_Intervals
END
```

List. 1. Interval Matrix Generation

Our algorithm in List. 1 is taking our very specific case into account, but it is definitely possible to make modifications to this algorithm to make it more generalized. Instead of only supporting days, support for other interval lengths like hours, minutes, or weeks could also be added. There are also certainly more complicated and/or more efficient ways to handle this. For example, one could also just create an ordered list of all possible minimal length intervals and then just combine subsequent intervals when necessary. For the purposes of this experiment efficiency is not an important factor, so it was decided to go for a simpler, more resource intensive algorithm.

Finding a Final Solution to the Query Once we have used our matrix to makes request using the sub-intervals, we can process that information and provide our answer to the initial query. In List. 2 we take a valued version of our matrix created in List. 1 and make use of Allen's Interval Algebra to find a set of intervals that cover the interval in the original query. It is worth noting that this purposefully abstracts exactly what "value" means here. The selection of the ideal interval is not our focus, rather the application of Allen's Interval Algebra.

```
INPUT:
  matrix of valued sub-intervals 2D_Matrix_of_Val_Intervals
```

```
3  OUTPUT:
4    set of intervals covering interval from Start_Date to End_Date
5
6  BEGIN
7    NEW Set Set_of_Intervals = {}
8    NEW Boolean Interval_Covered = FALSE
9
10   WHILE(NOT Interval_Covered):
11     Interval Most_Val_Interval =
12       GetMostValuableInterval(2D_Matrix_of_Val_Intervals)
13
14     Set_of_Intervals.add(Most_Val_Interval)
15
16     # remove all intervals that conflict with Most_Val_Interval
17     FOR i IN Range(0, Length(2D_Matrix_of_Val_Intervals)):
18       2D_Matrix_of_Val_Intervals[i] =
19         2D_Matrix_of_Val_Intervals[i].filter(
20           LAMBDA Interval interval:
21             interval < Most_Val_Interval OR
22             interval > Most_Val_Interval OR
23             interval m Most_Val_Interval OR
24             interval mi Most_Val_Interval
25         )
26     ENDFOR
27
28     # clean up matrix after removing conflicting intervals
29     RemoveEmptyListsInMatrix(2D_Matrix_of_Val_Intervals)
30
31     # if our matrix is empty, our set must completely cover
32     # the interval supplied during the initial query
33     IF Length(2D_Matrix_of_Val_Intervals) == 0:
34       Interval_Covered = TRUE
35
36   ENDWHILE
37
38   RETURN Set_of_Intervals
39 END
```

List. 2. Final Query Solution

In the algorithm defined in List. 2 lines 21-24 are our application of Allen's Interval Algebra. Given a selected interval, we filter out the intervals that conflict with it. There are two main cases here: 1. an interval takes place entirely before or entirely after the selected interval 2. an interval meets the selected interval either at its beginning or end point . These two cases are represented by the interval relations $<$, $>$, m, and mi. With all other relations it is compulsory that the intervals have some kind of overlap. The filter function used here is a function that removes any elements that do not return true using the supplied function (here a lambda function is used). This will result in some “rows” in our matrix being empty, which are then removed. This is actually optional, but is a small and simple optimization, which is why it was added. This also makes our final check as easy as checking for an “empty” matrix. Alternatively, we could use our interval algebra to check that our interval set (when combined into a single interval) is equal to our initial, user supplied interval.

3.5 Implementation Notes

With the discussion of the experiment algorithm concluded, I would like to add a few notes about the implementation. The programming language used for this experiment was Python. The library used for Allen’s Interval Algebra was IntervalAlgebra by Alex Riley [20]. This simple library written in C provides a base implementation that also natively works with Python’s DateTime classes. Finally, the API that was used to retrieve data on hotels was the APIDojo TripAdvisor API on RapidAPI [1]. No first-party hotel booking APIs that allowed use for the purpose of research were found, which is why this API was selected instead. This TripAdvisor API does not prohibit use in our work and it provides real results with the level of detail that is required for this experiment.

4 Experiment Results

4.1 Parameters

Before we look at the results of our experiment, I will provide a bit of explanation on the parameters used. At the time that this experiment was run, many countries in the world were experiencing lock-downs and tourism restrictions due to COVID-19. Given that we used a hotel use case, this is relevant to these results. This led to me choosing a location that was currently in an above average state in relation to the pandemic, Singapore.

To have results that may be more representative of real life cases, I decided to then work with two variables, how far in the future the stay should begin and how long its duration should be. Having examples that are short notice bookings as well as ones that are made far ahead of time seemed realistic. To simulate this, I chose starting dates approximately one week from the experiment date and starting dates approximately three months in the future. In addition, I considered what would be realistic for trips that are more tourism and trips that are more business. In the case of tourism, a weekend trip, arriving Friday and leaving Monday, is what I decided on. In the case of business, a week-long trip, I decided to use trip dates starting on Sunday and ending Saturday.

Finally, in terms of setting a value for hotel stays during a specific interval, I used a very simple metric. The TripAdvisor API [1] that I used usually returned a min and max price for a specific property during the days requested. A request would return a max of 30 properties. In the requests made during the experiment, it was also specifically set to return the cheapest properties available, as this was a way to bring some consistency to the results. I set the value for that interval as the average of the min and max for all properties. The metric used is not necessarily important and admittedly that there are certainly better metrics that could be used, but it fulfills the role that was needed to run the experiment.

4.2 Comparison: Base vs. Enhanced

For each of the four queries, I am presenting the results for the unmodified query as well as the Allen's Interval Algebra enhanced query. These results for each query come from requests that were made during a singular run of the program. The combined price average for the enhanced answers take into account the duration of the stay, as the average is calculated as a per day value.

Table 3. Short notice: 3-night weekend trip

	Interval	Price Average (per day)
Base	2020.11.06 - 2020.11.09	145.87
Allen's Interval	2020.11.06 - 2020.11.08	51.88
Algebra Enhanced	2020.11.08 - 2020.11.09	148.52
	combined	84.09

Table 4. Advance notice: 3-night weekend trip

	Interval	Price Average (per day)
Base	2021.02.05 - 2021.02.08	143.04
Allen's Interval	2021.02.05 - 2021.02.06	51.16
Algebra Enhanced	2021.02.06 - 2021.02.08	50.79
	combined	50.91

Table 5. Short notice: Week-long trip

	Interval	Price Average (per day)
Base	2020.11.08 - 2020.11.14	58.40
Allen's Interval	2020.11.08 - 2020.11.09	57.16
Algebra Enhanced	2020.11.09 - 2020.11.11	51.07
	2020.11.11 - 2020.11.13	49.93
	2020.11.13 - 2020.11.14	146.31
	combined	67.58

Table 6. Advance notice: Week-long trip

	Interval	Price Average (per day)
Base	2021.02.07 - 2020.11.13	53.00
Allen's Interval	2021.02.07 - 2020.11.08	53.00
Algebra Enhanced	2021.02.08 - 2020.11.09	146.09
	2021.02.09 - 2020.11.12	48.75
	2021.02.12 - 2020.11.13	146.09
	combined	81.91

Analysis The results are overall very positive from the perspective of showing the value of Allen's Interval Algebra for this type of application. For both of the 3-night weekend trip queries, the enhanced version provided results that were on average lower than the price for the entire interval. Even in the case in Table 3 where a part of the result was on average higher than the base result, the savings in the other part of the result were large enough to make the overall average much lower.

The results for the week-long trip queries were less conclusive. While the short notice version in Table 5 provided an answer where 5 of the 6 parts of the interval were below the average of the base query's result, the final interval was much higher and significantly increased the average. For the advance notice result in Table 6, the results could be seen as significantly worse. This is actually not much of a surprise though as with a query for a hotel so far into the future, having to deal with a limited hotel offering is less likely.

The goal of this experiment was to show that enhancing the understanding of temporal elements in a query can provide benefits. These results are conclusive enough to say that a benefit is provided. This experiment and its results are not without criticism though.

Specifically in these results, both of the week-long solutions showed a possible place to improve our algorithm. A very simple ordered list of intervals was used and conflicting intervals were removed after each interval selection. This could very easily have lead to a situation where an overall cheaper combination was removed because one part of a more expensive combination was first in the queue. Again, this does not disqualify our results, rather it provides an interesting place to put more consideration into how to more intelligently solve the problem.

5 Criticisms

In addition to the criticisms based on the results of our experiment, there are a few criticisms of this approach as a whole. A few of them have already been mentioned over the course of our discussions in this article. Before we conclude, I would like to take a bit more time to address them here. The three I would like to further discuss are: resource usage, value determination, and solution methods.

5.1 Resource Usage

One of the things about the approach presented here that is certainly a detriment is that it is overall much more resource intensive than answering queries without enhanced reasoning. For one, in this implementation we are creating objects in the order of $\sum_{n=1}^{x-1} n$, where x is the duration of the temporal element divided by the desired minimum interval length. Depending on the data that needs to be stored in each interval, the lengths of intervals desired, and the hardware handling this request, this can be significant. It is also possible to set a min or max length of interval, which would lower resource demand. These limitations were not something that was considered for this experiment.

In addition to the memory requirements, we also are making requests for each of our sub-intervals. For each of the intervals, a few translations and calculations are run on them. There is without a doubt a more significant cost here. However, this was also something I specifically decided not to address in this work. There are ways that this could be improved if more research was done into the specific use case. This may actually also be an unnecessary concern in many cases. In our experiment, no performance issues related to the increased data or calculation load were observed. Modern devices like smart phones or even edge computing devices are likely fairly capable of handling many cases.

5.2 Determining Value

One of the points in this experiment and results that can be easily criticized in our method for determining value. The argument still stands that this was not a focus, but that does not mean it is not worth of a discussion here. I chose a simple and easily calculable metric, but there is nothing stopping a develop from producing a more complicated value function. This is something I tried to address early with the separation of the value determining function from the actual selection of the intervals in the final answer.

5.3 Solution Methods

In addition to the value function, there is another place where we used a very simple approach. It is fairly clear with the results of Table 5 and Table 6, we did not find a better solution. This can be attributed to a failure in our value function or in our algorithm as a whole. We look for and return a single solution, but we knew from the beginning that this would not necessarily be optimal. There are potential ways to get around this, like doing an exhaustive search for the best solution or finding a set of solutions and taking the best with the knowledge that it is possibly not optimal either.

5.4 Future Work

These three criticisms, while entirely valid, do not necessarily invalidate our results. Our goal, once again, was to show that temporal reasoning can be helpful and I still think this is the case. What these criticisms do lead us to is a very good direction for further research into this field. These three are topics that would have to be addressed before any true adoption of this work. Also of great importance would be an overall more robust usage of Allen's Interval Algebra. I specifically implemented our example with only day-long intervals in mind, but there could certainly be other applications that make use of minute-long, hour-long, or any other duration length intervals. I believe that this work shows promise and that these topics would provide interesting contributions.

6 Outlook and Conclusion

6.1 Outlook

The possible applications of Allen's Interval Algebra is something that was partially addressed in Section 3. Using it for Smart Home/Office or IoT applications is certainly possible. There are also possibly other applications in the world of digital personal assistants that we never even considered. This work also only used a small subset of the relations that Allen defined. Another possible application could be helping with scheduling, possibly in tandem with IoT. A large office with many meeting rooms could also use the power of digital assistants enhanced with Allen's Interval Algebra to book meeting rooms more efficiently.

One of the most exciting things about digital personal assistants that this point in time is that they are still fairly raw in terms of their base abilities, but adoption is increasing, so there are many more chances to find an audience for more "niche" applications. "Niche" would be a reasonable way to interpret our use case here and although it might appear useless to many, to a very specific set of users it may seem to be a very helpful alternative. Regardless of the opinion on our use case, providing the tools presented here should be considered. If that is done, there are other likely developers out there that would see these tools and come up with a new idea for a useful service that they otherwise might not have considered.

6.2 Conclusion

In this work we considered the field of digital personal assistants. We analyzed them and found a possible direction for future improvements in their base abilities, specifically improvements to reasoning. We considered some types of reasoning and decided to address a possible application of temporal reasoning. We conducted our experiment and our results showed that additional temporal reasoning abilities could be beneficial in providing the solution to a type of query that the user of a digital assistant may ask. The work presented here is not without faults, but it presents a potentially interesting future direction for smart assistants and potentially some other applications. With that in mind I can say that this work accomplishes the goal that we set out to reach in the beginning.

2

Appendix

A Advertised Alexa Commands

Table 7. Alexa Commands [4]

Command	Type	Category
Alexa, try Amazon Music	task	Entertainment
Alexa, play baby shark	task	Entertainment
Alexa, play the new Katy Perry album on Apple Music	task	Entertainment
Alexa, play the song that goes 'love is all you need'	task	Entertainment
Alexa, set an alarm to Adele every day at 10 AM	task	Productivity
Alexa, play Z100	task	Entertainment
Alexa, play the latest Homecoming podcast	task	Entertainment
Alexa, what free audiobooks are available?	query	Entertainment
Alexa, what's on TV tonight?	query	Entertainment
Alexa, what is Drop In?	other	Communication
Alexa, how do I call people?	query	Communication
Alexa, Drop In on the kitchen	task	Communication
Alexa, call Mom	task	Communication
Alexa, Drop In on Grandpa's Echo	task	Communication
Alexa, how do I make announcements?	query	Communication
Alexa, answer	task	Communication
Alexa, call Jeff's Echo	task	Communication
Alexa, announce that we're leaving in five	task	Communication
Alexa, announce that I'll be home by 6:30	task	Communication
Alexa, set a recurring alarm for 7:00 AM	task	Productivity
Alexa, what's on my calendar for today?	query	Productivity
Alexa, remind me to call mom on Saturday at 2:00 PM	task	Productivity
Alexa, reply to this email	task	Productivity
Alexa, what's 100 dollars in Euros?	query	Knowledge
Alexa, add orange juice to my shopping list	task	Productivity
Alexa, spell superfluous	task	Knowledge
Alexa, find me a recipe for banana bread	task	Knowledge
Alexa, remind me to get groceries when I get home	task	Productivity
Alexa, schedule a meeting with Jeff	task	Productivity
Alexa, turn on the kitchen lights.	task	Smart Home/IoT
Alexa, dim the bedroom lights.	task	Smart Home/IoT
Alexa, turn the living room lamp blue.	task	Smart Home/IoT
Alexa, turn on the lamp.	task	Smart Home/IoT
Alexa, turn off the fan.	task	Smart Home/IoT
Alexa, turn on the coffee maker.	task	Smart Home/IoT
Alexa, set the living room to 72.	task	Smart Home/IoT
Alexa, what's the temperature in here?	query	Smart Home/IoT
Alexa, turn the temperature down.	task	Smart Home/IoT
Alexa, show me the nursery.	task	Smart Home/IoT
Alexa, show me the front door.	task	Smart Home/IoT

(Table 7 continues on the next page)

Command	Type	Category
Alexa, reorder paper towels.	task	Shopping
Alexa, search for dog toys.	task	Shopping
Alexa, find the best coffee maker.	task	Shopping
Alexa, what are the best-selling toys?	query	Shopping
Alexa, what are my deals?	query	Shopping
Alexa, how much is an Instant pot?	query	Shopping
Alexa, what are my Whole Foods deals?	query	Shopping
Alexa, add bananas to my Whole Foods cart.	task	Shopping
Alexa, where's my stuff?	query	Shopping
Alexa, what are my notifications?	query	Productivity
Alexa, what's the weather tomorrow?	query	Knowledge
Alexa, what's in the news today?	query	Knowledge
Alexa, what's 98 divided by 26?	query	Knowledge
Alexa, how do you say 'I love you' in Japanese?	query	Knowledge
Alexa, what's the score of the Mariners game?	query	Knowledge
Alexa, how many people are on Earth?	query	Knowledge
Alexa, tell me a joke	other	Specialty
Alexa, good morning!	other	Specialty
Alexa, tell me a math joke	other	Specialty
Alexa, what's the shortest word in the dictionary?	query	Knowledge
Alexa, give me an Easter egg	other	Specialty
Alexa, tell me a poem	other	Specialty
Alexa, give me a hard-boiled Easter egg	other	Specialty
Alexa, tell me a love story	other	Specialty
Alexa, how are you?	other	Specialty
Alexa, sing a campfire song	other	Specialty
Alexa, what's your favorite color?	other	Specialty
Alexa, what's your favorite number?	other	Specialty

B Advertised Google Assistant Commands

Table 8. Google Assistant Commands [9]

Command	Type	Category
Set a 15-minute timer for cookies	task	Productivity
Remind me to drink water every morning	task	Productivity
Add eggs and bread to my shopping list	task	Productivity
What's on my shopping list?	query	Productivity
Remember my Wi-Fi password is 987654	task	Productivity
Set an alarm for 7 AM	task	Productivity
Remind me to do laundry when I get home	task	Productivity
Remember where I parked	task	Productivity
Call Dad	task	Communication
Call my voicemail	task	Communication
Send a text	task	Communication
Read my unread texts	task	Communication
Email Mom	task	Communication
Read my new emails	task	Communication
Text Sam 'On my way'	task	Communication
Call Grandma on speakerphone	task	Communication
What's the weather right now?	query	Knowledge
How's the traffic to work?	query	Knowledge
Give me directions to the airport	task	Knowledge
Find the closest ATM	task	Knowledge
What time does the post office close?	query	Knowledge
Call the nearest pharmacy	task	Productivity
Will it rain tomorrow?	query	Knowledge
Find movies playing nearby	task	Knowledge
How many ounces are in a pound?	query	Knowledge
What's 20% of 47?	query	Knowledge
How do you say hello in Chinese?	query	Knowledge
How much protein is in an egg?	query	Knowledge
What time is it in London?	query	Knowledge
What's on my schedule today?	query	Productivity
When is sunset?	query	Knowledge
What is the S&P 500 trading at?	query	Knowledge
Play workout music	task	Entertainment
Play Today's Top Hits on Spotify	task	Entertainment
Tell me the latest news	task	Entertainment
Play NPR news summary	task	Entertainment
Listen to ESPN SportsCenter	task	Entertainment
Play rain sounds	task	Specialty
Listen to Hidden Brain	task	Entertainment
Set volume to 3	task	Productivity
Let's play a game	other	Specialty
Tell me a joke	other	Specialty
What sound does a cow make?	other	Specialty
Let's play Lucky Trivia	other	Entertainment
Sing Happy Birthday	other	Specialty
Tell me a fun fact	other	Specialty
Play Freeze Dance	other	Entertainment
Read me a poem	other	Specialty

C Advertised Siri Commands

Table 9. Siri Commands [15]

Command	Type	Category
Call Mom on speaker	task	Communication
Text Donna “I’m on my way exclamation point”	task	Communication
Read my last message	task	Communication
Message Teri “I’ll be there in 30 minutes” on WhatsApp	task	Communication
FaceTime Ivy	task	Communication
Wake me up at 7 am	task	Productivity
Split a $94 check three ways	task	Productivity
Remind me to pick up the dry cleaning when I leave work	task	Productivity
What does my day look like?	query	Productivity
Set a timer for 20 minutes	task	Productivity
Play me something I’d like	task	Entertainment
What song is this?	query	Entertainment
Play the latest Vagabon (specific artist)	task	Entertainment
Play me some 90s hip hop	task	Entertainment
When did this song come out?	query	Knowledge
Turn on the lights in the living room	task	Smart Home/IoT
I’m home	other	Specialty
Warm it up in here	task	Smart Home/IoT
What did they say? (while watching TV)	query	Smart Home/IoT
Show driveway camera	task	Smart Home/IoT
Did I close the garage door?	query	Smart Home/IoT
Watch His Dark Materials	task	Entertainment
How do you say “Thank you” in Mandarin?	query	Knowledge
How many dollars is 45 euros?	query	Knowledge
How are the reviews for Booksmart?	query	Knowledge
What’s the derivative of cosine(x)?	query	Knowledge
What’s the score of the Clippers game vs the Spurs?	query	Knowledge
Show my photos from Thailand last summer	task	Entertainment
Find my AirPods	other	Smart Home/IoT
Search Pinterest for photos of kitchens with white subway tile	task	Knowledge
Open the presentation I was working on yesterday	task	Productivity
Apple Pay Adam $50	task	Productivity

3

Bibliography

References

1. Tripadvisor api (????), https://rapidapi.com/apidojo/api/tripadvisor1
2. The team behind Siri debuts its next-gen AI "Viv" at Disrupt NY 2016. TechCrunch (May 2016), https://www.youtube.com/watch?v=MI07aeZqeco
3. Actions on google (2020), https://developers.google.com/assistant
4. Alexa features (2020), https://www.amazon.com/b/ref=aeg_lp_features/ref=s9_acss_bw_cg_aeglp_md1_w?node=17934672011
5. Allen's interval algebra (Jun 2020), https://en.wikipedia.org/wiki/Allen's_interval_algebra
6. Amazon alexa (Oct 2020), https://en.wikipedia.org/wiki/Amazon_Alexa
7. Getting started with the alexa skills kit (2020), https://developer.amazon.com/en-US/alexa/alexa-skills-kit/start
8. Google assistant (Oct 2020), https://en.wikipedia.org/wiki/Google_Assistant
9. Google assistant - learn what your google assistant is capable of (2020), https://assistant.google.com/learn/
10. Google nest (Oct 2020), https://en.wikipedia.org/wiki/Google_Nest
11. Google now (Oct 2020), https://en.wikipedia.org/wiki/Google_Now
12. Headspace: Guided Meditation for Everybody (Apr 2020), https://build.amazonalexadev.com/rs/365-EFI-026/images/RAIN_Headspace_Final05082020.pdf
13. Mad Libs brings their best-selling games to Google Assistant with Interactive Canvas (2020), https://developers.google.com/assistant/downloads/mad-libs-casestudy.pdf
14. Siri (Oct 2020), https://en.wikipedia.org/wiki/Siri
15. Siri (2020), https://www.apple.com/siri/
16. Sirikit (2020), https://developer.apple.com/siri/
17. Allen, J.F.: Maintaining knowledge about temporal intervals. Communications of the ACM 26(11), 832–843 (Nov 1983)
18. Mierzejewski, M.: Digital assistants, the next marketing revolution (Dec 2018), https://www.thinkwithgoogle.com/marketing-strategies/app-and-mobile/digital-google-assistant/
19. Olson, C., Kemery, K.: Voice Report (Apr 2019), https://advertiseonbing-blob.azureedge.net/blob/bingads/media/insight/whitepapers/2019/04%20apr/voice-report/bingads_2019_voicereport.pdf
20. Riley, A.: Intervalalgebra (Jan 2019), https://github.com/ajcr/IntervalAlgebra

All links were last followed on October 31, 2020.