The applicability of using quantum computing to solve geospatial optimization problems

Preface

The OGC has been defining standards in a spatial context for over 20 years. As new technologies emerge and become relevant to solving spatial problems, the need for standards to integrate these new technologies into existing enterprises is crucial. This, coupled with ensuring data and services are FAIR (Findable, Accessible, Interoperable, Reusable) provides a roadmap for rapid technological integration. Prior to standardization, it is prudent to understand the applicability of new technologies to OGC concerns, which in this context is geospatial optimization problems. This document describes the emerging technology of quantum computing through a short background and history followed by the practical implementation of three use cases. Future work will start to look at the case for a formal interface, data model, or best practice for using quantum computing in a geospatial enterprise for solving optimization problems.

Abstract

This discussion paper provides an overview of the emerging technology of *quantum computing with a focus on its applicability to solving geospatial problems. The advantages of using a computer built upon the properties of quantum mechanics to some complex (NP-hard) problems has been known since the mid-1990s. However, the engineering challenges with building quantum computers have meant that quantum computers of sufficient size to solve practical problems have only recently become available. Although there are many applications that may benefit from using a quantum computing approach, this*

paper focuses on optimization problems as there are many within the geospatial domain, however, optimization problems are best solved by adiabatic quantum computing which is a specialized machine that is setup to only solve optimization problems. Three use cases are discussed and run on a cloud connected D-Wave quantum computer; the traveling salesperson problem, structural imbalance in a social network, and optimization of satellite constellations according to a cost/benefit balance. The findings are that practical quantum computing for geospatial analytics is practical, useful, and through the use of hybrid samplers can provide real-world advantages over classical computing.

Introduction

This discussion paper is concerned with the topic of quantum computing (QC) and its applicability to addressing geospatial problems. The Open Geospatial Consortium (OGC) is an innovative, member led organization concerned with creation and maintenance of standards for geospatial data, processing, and adjacent technologies. It has been operating for over 20 years with a guiding principle that geospatial assets should be FAIR (Findable, Accessible, Interoperable, Reusable). Quantum computing is of interest to the OGC because of its applicability to uniquely complex, geospatial problems.

The focus of the experimentation described in later sections was *integration* and *experimentation* rather than discerning new quantum algorithms or looking at generation after next capabilities. Part of the motivation was the pace of technological change and specifically the rise of technologies such as ChatGPT and LLMs in general which came to public prominence almost overnight and answering the following questions:

- What if QC has it's *ChatGPT moment?*
- What could QC be used for in the geospatial domain?
- What, if any, are the advantages of using QC in the geospatial domain either now or in the future?
- What is the integration path for QC with existing geospatial technologies?

Although quantum technologies are not new, the recent interest in QC for computationally difficult applications is notable with many large technology companies spinning up quantum divisions in both the applications and engineering specialisms. Additionally, there are specialist companies working on engineering QCs, with further private entities working on applications. This paper provides a discussion on applications and potential applications for quantum computing, although there is an introduction to some of the concepts and challenges with QC it is largely focused on the practical applications of QC for geospatial problems but does provide a cursory discussion of the approaches to QC and the challenges with the engineering and hardware. As with many technologies, QC is constantly evolving in terms of capabilities, speed and overall hardware.

The promise of QC has been understood prior to any hardware implementations. Perhaps the most well-known (and for some, concerning) applications is the ability for QCs to factor very large numbers. Although this may seem a trivial application for these machines, many encryption protocols are built upon the principle that deriving prime factors from a very large number is computationally hard, although not impossible. A salient example is the use of public and private keys for securing pieces of information where both the public and private key are required to decrypt information and read the contents. Public keys are transmitted with the understanding that they maybe intercepted, but because of the nature of computing factors from large numbers using classical computing, it is computationally hard to derive the private key with knowledge of the public key. However, generating the public key from the private key is computationally trivial. Although current QCs are not able to crack this type of encryption with current, small to medium sized machines, encrypted information can be collected with the ambition of decrypting it in the near future. Interestingly, the ability for QC to factor large numbers was proven in the 1990s by Peter Shor (Shor's algorithm) long before practical implementations of QCs.

Another often cited QC application is searching of large databases. This application was investigated by Grover who produced an algorithm. As with Shor's algorithm, a quantum advantage to searching a database was shown albeit quadratic rather than exponential. An additional consideration for practical implementation is the reliance on an *oracle function*, which somehow derives what the result of the search should be and the QC locates the record or records in the database.

There are several other applications that are gaining traction for QC use cases including interaction with sensors and *quantum machine learning*. At time of writing, the ML applications do not have the same level of *proof* as the previous examples, but have shown promise.

Quantum Computing

Classical computers work using *bits*. Bits are the fundamental building blocks of computation that can have one of two states at any one time (usually styled as 0 and 1 or *off* and *on*). QC is different in that its building blocks are *qubits*, which exist in a superposition of 0 and 1. Often this is visualized as *Bloch Sphere* where a spin upwards is analogous to *0* and a spin downwards is *1*. Any other spin is a superposition of 0 and 1 because it contains components of both. This ability to hold a superposition of qubits is fundamental to QC approaches.

A second component of QC is *entanglement*, which was famously dismissed by Einstein as "spooky action at a distance." Entanglement happens when two particles are linked and can effect each other's state regardless of their distance from each other. This also has implications for quantum teleportation and information exchange, although this does not cause issues with causality because information regarding the state change on one particle is required to get information out of the second particle and the transmission of the information must be done classically with all of the speed of light restrictions associated with it.

An additional consideration of QC (of which there are many) is that the process is non-deterministic, unlike classical computing that will produce the same result every time provided the initial conditions are identical, QC can produce different results with the same input. A method of addressing this issue is to perform the calculation many times (1000s) and averaging the results in some way (usually the mode if talking energy states).

Types of quantum computing

QC implementations have emerged from large technology companies as well as small and middle size disruptive businesses looking to leverage practical QC as the technology evolves. Unlike *classical computing* which is generalized, QC is specialized to solve certain types of problems. Although QC shows promise for some applications, it is not currently envisaged that QC or Quantum Processing Units (QPUs) will replace classical computers, this is not like the invention of the microchip replacing the valve approach to generalized computation! Currently there are two conceptual approaches to QC, these are:

- Circuit based.
- Adiabatic or annealers.

Circuit based QC are potentially closer to generalized classical computing. These are the machines that can factor large numbers and perform search algorithms as described in the introduction. Indicatively, this type of QC is where much of the focus of the research community lies, this maybe because it has the potential to offer quantum advantage, that is, it can solve problems that cannot be solved in a sensible time by a classical computation approach (note that this includes the world's most powerful supercomputers).

Adiabatic quantum computing (AQC) is a different type of computation that is setup specifically for solving optimization problems. The hardware and topology of AQC is completely different from circuit based approaches, which means that unlike generalized classical computing, you cannot run circuit based algorithms on AQCs as they way that they perform computation is fundamentally different. Simplistically, whilst circuit based QC is concerned with the manipulation and transformation of individual qubits, ADQ looks to put the entire system in a very low energy state and then transfer the problem space slowly into the AQC with the result to the entire problem emerging from maintenance of the low energy state. One way to think about this is the machine state starts off representing a horizontal plane, that is, it is flat. After the problem has been transferred into the ADQ, the plane is no longer flat, instead it has a topology, much like a hilly landscape. The optimal solution of the problem space is the deepest valley such that if a ball was dropped onto this hypothetical landscape, the *ball* would end up at the lowest point in the landscape which represents the solution. Computationally, this means that the machine needs to *find the ball* to get the solution.

There are many online resources that provide a fuller explanation including the physics of QC and quantum information theory, but it is beyond the scope of this discussion paper that is focused on applications.

Quantum hardware and engineering challenges

The hardware and engineering challenges with QC have contributed to its relatively slow progress compared to other technologies such as artificial intelligence, or the increase in power of graphical processing units (GPUs). Two big challenges with QC hardware are *noise* and *decoherence*. Qubits are fragile and cannot hold state for long periods of time. There is also the fundamental problem of interacting with a system that for all intents and purposes needs to be isolated from the outside world to maintain coherence. One method of addressing these issues is to increase the number of physical qubits to create a smaller number of *logical* qubits where the outputs from each bank of

physical qubits is corrected. The result of this approach is that it makes quantum computation far more reliable that those without the correction, however, it requires many times more physical qubits to produce a number of logical qubits that can do computation of practical and useful size.

From a hardware perspective, there are different approaches to creating physical machines and addressing the challenges of noise and decoherence. Some examples are:

1. Superconducting approaches - where materials are cooled to close to absolute zero to produce quantum effects.
2. Trapped ions - ions trapped using electromagnetic fields.
3. Photons - using particles of light to perform computation.
4. Neutral atoms - use of lasers to arrange atoms into grids.
5. Annealers - use of phenomenon *quantum tunnelling* to find the minima of a function for optimization problems (the main concern of this paper with respect to geospatial applications).

There are other methods being actively investigated, however, the number of approaches demonstrates that experimentation is still being conducted, and the *best* technology has yet to be settled on. This is in contrast to classical computing where silicon and to a lesser extent copper are the metals used to create microchips.

Overall, these technologies represent engineering challenges. Unlike CPUs which reside locally inside classical computers, it appears unlikely that a QPU will sit along side CPUs like GPUs do, albeit in the cloud considering the size of current quantum computers (square metres in size). The current generation of QCs are of a similar size to the early classical mainframes and therefore require infrastructure to host locally. The main method of accessing quantum computation today is to use cloud connected services.

Interaction with quantum computers

Creating and executing jobs for todays quantum computers is done through software development kits (SDKs). As QPU time can be expensive, the SDKs often offer a *quantum simulator* to test code locally with the quantum output simulated via a classical computer. Some SDKs, especially for circuit based machines offer the ability to create and simulate quantum circuits. Individual qubits are tasked within the circuit to produce a result; although this is useable for the size of QCs that are available now, it will quickly become impractical with even a small number of qubits. Many of these SDKs offer the ability to call a QC with parameters to execute a known computation without having to do low-level programming, a salient example of this is with quantum annealing where the problem space is restricted to optimization problems. Some examples of SDKs are as follows:

1. Qiskit - IBM - used for constructing and executing circuits.
2. Cirq - Google - Python library tailored to Google's quantum machines.
3. Ocean Software - used for quantum annealing on D-Wave's machines.
4. Microsoft Quantum Development Kit - uses Q#, a language for expressing quantum algorithms.

There are also many more SDKs with some intending to be cross platform.

Much of the work described in this paper was conducted using the Ocean Software SDK to execute D-Wave cloud connected quantum computers.

Geospatial Applications

The OGC is a longstanding and premier standards body for the geospatial industry. With over 20 years of history, the OGC has been a thought leader in emerging technologies and whose standards have facilitated interoperability across IT enterprises and seen wide adoption in industry leading products and governmental organizations.

As discussed in previous sections, the geospatial domain has many problems that maybe addressable by QC. For example, geospatial data has always been *large*, long before the concept of *big data* and finding items with fast search algorithms has been a topic of interest for some time, especially when spatial operators are involved which increases the complexity. Beyond search is the concept of *optimization*, which is broadly the solving of complex computations to find the *best* solution given a very large number of potential solutions. From a geospatial perspective this includes the following:

1. Routing - given the possibility of many routes, which is the optimal?
2. Optimal location planning - given a list of possible locations, where should I place my services to give the largest number of people access within a time constraint?
3. Geospatial relationships - given a list of locations, people, and relationships between them what patterns can be discerned?
4. Knowledge graphs - given a number of entities and relationships, what is the optimal path to get from A to B given C and d?
5. Orbital mechanics and communications - what is the optimal constellation of satellites given these constraints?
6. Data reduction and sampling. Given the wealth of data available (data deluge), can datasets be sampled to remove as much as possible whilst maintaining the hetrogeny and production of quality downstream outputs?

Optimization problems are often transferrable to other, adjacent problems, and because many of these problems are well known and reducible to other problems, the programming required to express them in quantum terms is often reduced to a call to a server with parameters rather than starting from first principles.

The following sections outline the general approach to AQC for solving optimization problems and some practical use cases that have been tested by integration of AQC into current geospatial software capabilities.

Computational difficulty of optimization problems

Not all optimization problems are the same but they do have some commonalities, the main one

being that there are many solutions to the problems stated and finding the correct (optimal) one is the challenge because of the number possible solutions available. Many optimization problems have a factorial (noted as n!) component within them. This means that for every layer of complexity, node, destination, relationship, or variable added to the problem makes finding the solution exponentially more complex. An often cited example is the number of possible calculations in a standard deck of 52 cards, this is clearly 52!. The result of this calculation is 8.1x10^67 possible combinations of a deck of 52 cards, the result of this is that every single deck of cards that has every existed and will ever exist is likely a unique ordering of the deck (provided that it's shuffled correctly).

The *deck of cards* example shows that even with a relatively small number of nodes problems can quickly become unfeasible for the classical computing approach.

Optimization approaches

Binary Quadratic Models (BQMs) are a general method of mathematically expressing optimization problems; the D-Wave system uses these extensively. The quantum approach to using BQMs is to express variables as a binary (0 or 1) and then focus on optimizing quadratic functions of these variables. A BQM represents the energy state of a system with the QC tasked with minimizing or maximizing the energy state depending on the problem expressed. Like a BQM, which can manage constrained and unconstrained variables, a Quadratic Unconstrained Binary Optimization (QUBO) maybe used when the problem space is open, however, the same principle applies with the AQC tasked with minimizing or maximizing the energy state to produce the solution, which is represented by the state of the machine when the optimization is completed.

Use Cases

This section outlines three geospatially relevant optimization problems that have been tested using D-Wave's quantum annealer. Each use case contains a description of the problem, the approach to sending the problem to the AQC, the results returned and the issues encountered. The AQC was cloud based and could be executed provided the local machine had internet connectivity. For the first two use cases (TSP, SIB), plugins for Esri ArcGIS Pro were created to ensure integration of QC within a geospatial workflow. Additionally, each of the use cases described were created from examples on the D-Wave website as a starting point for the integration exercises [link::https://github.com/dwave-examples].

All of the problems stated are *at least* NP hard (from the famous P=NP consideration), with Traveling Salesperson Problem potentially being NP Complete according to some sources. This paper does not discuss the implications for NP completeness apart from to provide evidence that the optimization problems discussed are indeed *difficult* for classical computers.

Traveling Salesperson Problem

The Traveling Salesperson Problem (TSP) is typical in geospatial and logistics. There are different formulations, the one used in this experiment is as follows:

1. There is a salesperson with a known start location.

2. There are a number of locations that the salesperson must visit.
3. The solution is reached when the salesperson visits each of the locations once and only once in the lowest *cost* possible.

There are several ways to generate *cost* which may include:

- Straight line distance.
- Drive time.
- Road network distance.
- Efficient fuel usage.

Considering a *symmetrical* cost matrix, that is, point A to point B costs the same to travel between as point B to point A, the mathematical formulation of the problem is as follows:

stem::(n-1)!/2

An asymmetrical cost matrix would have an even larger problem space as it would not have the divide by 2 in the formula. The result of this formula is that:

- A 5 city problem has 12 possible routes.
- A 10 city problem has 181,440 possible routes.
- A 100 city problem has 4.7x10^155 possible routes.

As described, the number of possible routes quickly becomes unmanageable for classical computers to solve. Additionally this number of destinations maybe a problem with applicability in the real-world (considering large logistical operations such as Amazon). The problem becomes more challenging when the number of drivers also increases and the problem becomes allocation of drivers to deliveries *and* the routing concerns with TSP.

A caveat to this problem is that the comparison done is between the brute force version of the classical TSP algorithm (i.e., trying all of the combinations and picking the least cost). In logistics, there are many methods of solving this algorithm that are more efficient such as place aggregation, use of heuristics or other methods of reducing the problem to a manageable size. Therefore the overall utility of AQC in this space is left to the logistics domain experts to judge.

TSP Algorithm preparation

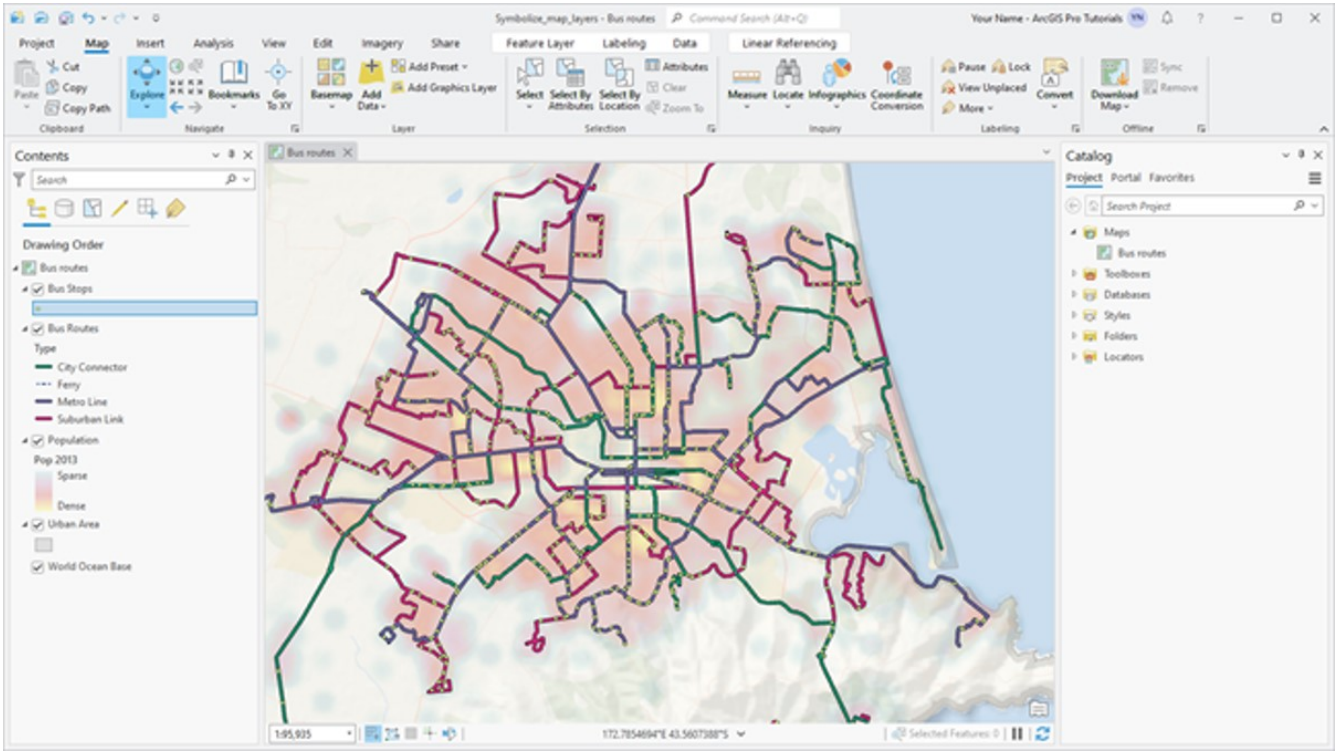
As TSP is a typical geospatial problem, D-Wave has a call that can be made to a quantum machine to solve TSP with the following parameters:

- A cost matrix.
- Configuration of the AQC (number of times to do the computation).

As part of the experiment two cost matrices were generated, one using straight line distance and one using the road network.

TSP Results

The AQC was able to quickly compute the correct route for 5 cities but often failed with 8 cities. As the ADQ is non-deterministic and has some noise in the system, it occasionally produces invalid results such as visiting the same destination twice. When compared to the classical method of doing TSP, the quantum approach is orders of magnitude quicker showing promise for the technology in this space.



Structural Imbalance within a Graph

Although not a strictly geospatial construct, the use of knowledge graphs in the geospatial domain and the OGC means that they can be used to represent geospatial information with one of the links between entities being locational *nearness* or otherwise. This use case uses geospatial analytics to contextualize the results of the quantum output. In future work, location could be used as an input

to this type of algorithm.

The Structural Imbalance Problem (SIP) is a special case of the *maximum cut* problem, which is a method of classifying graphs into two groups of vertices where the optimal solution is the one that maximizes the number of edges between the two groups. SIP seeks to classify graphs (usually social networks) using the rule "the enemy of my friend is my enemy". This is a simplification of real social networks, as there are many instances when individual relationships are more complex than *enemy* or *friend*, however it is useful for identifying those problematic relationships that go against how they *should* behave according to the model. Additionally, the geography of those relationships can provide some insight into regional events and trends.

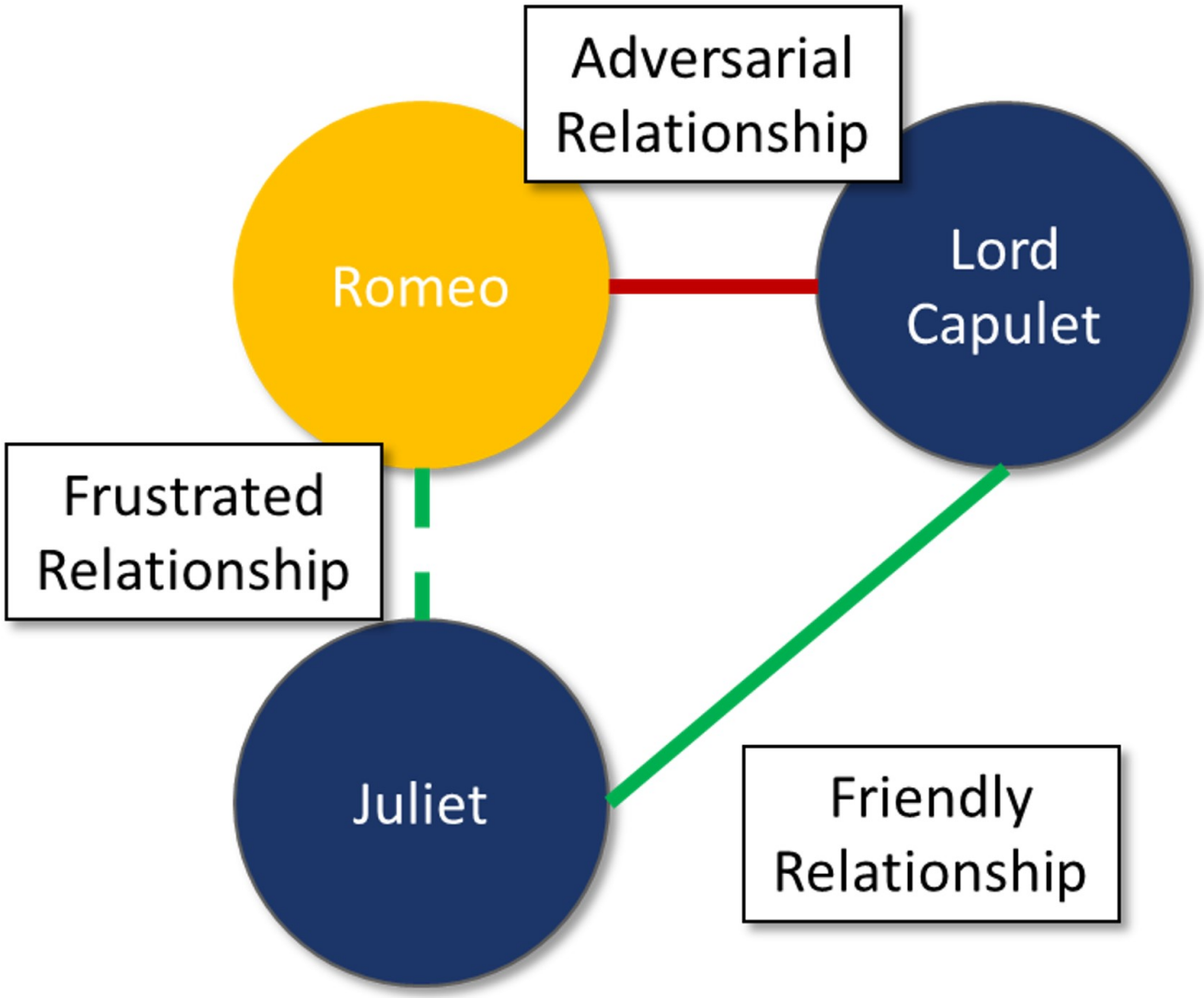
The mathematical formulation of SIP is as follows:

stem::Minimize ∑ (i,j)□E-wij xi xj

This involves minimizing the objective function according to the positive (friendly) and negative (adversarial) relationships between the entities in the graph. The algorithm can result in two sets of results:

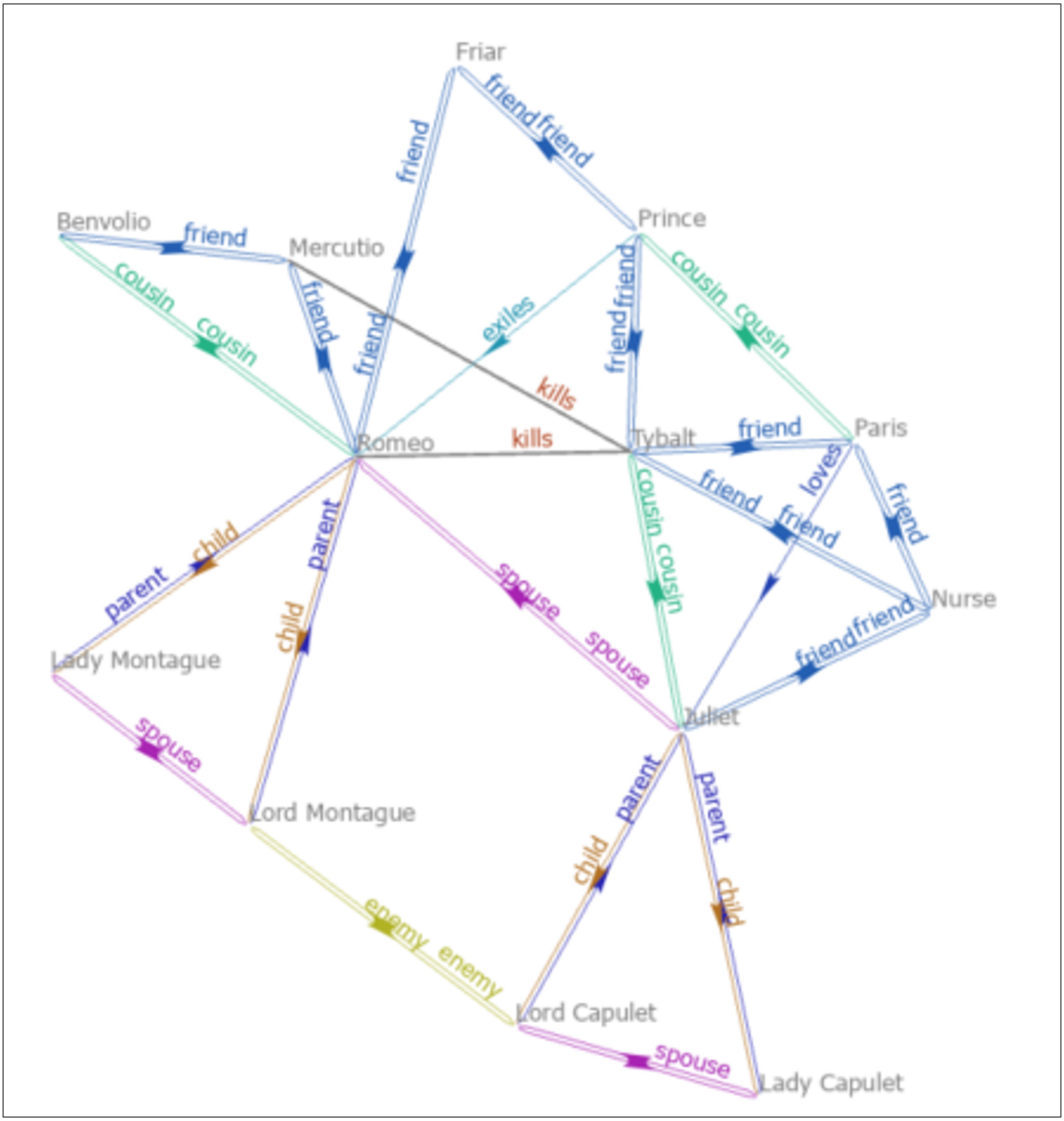
1. A perfectly balanced graph is one where all relationships between individuals within groups are friendly, and all relationships between groups are hostile.
2. An unbalanced graph is one where there are relationships within the graph that break the rule, that is, there are relationships that are friendly that should be hostile and hostile relationships that should be friendly.

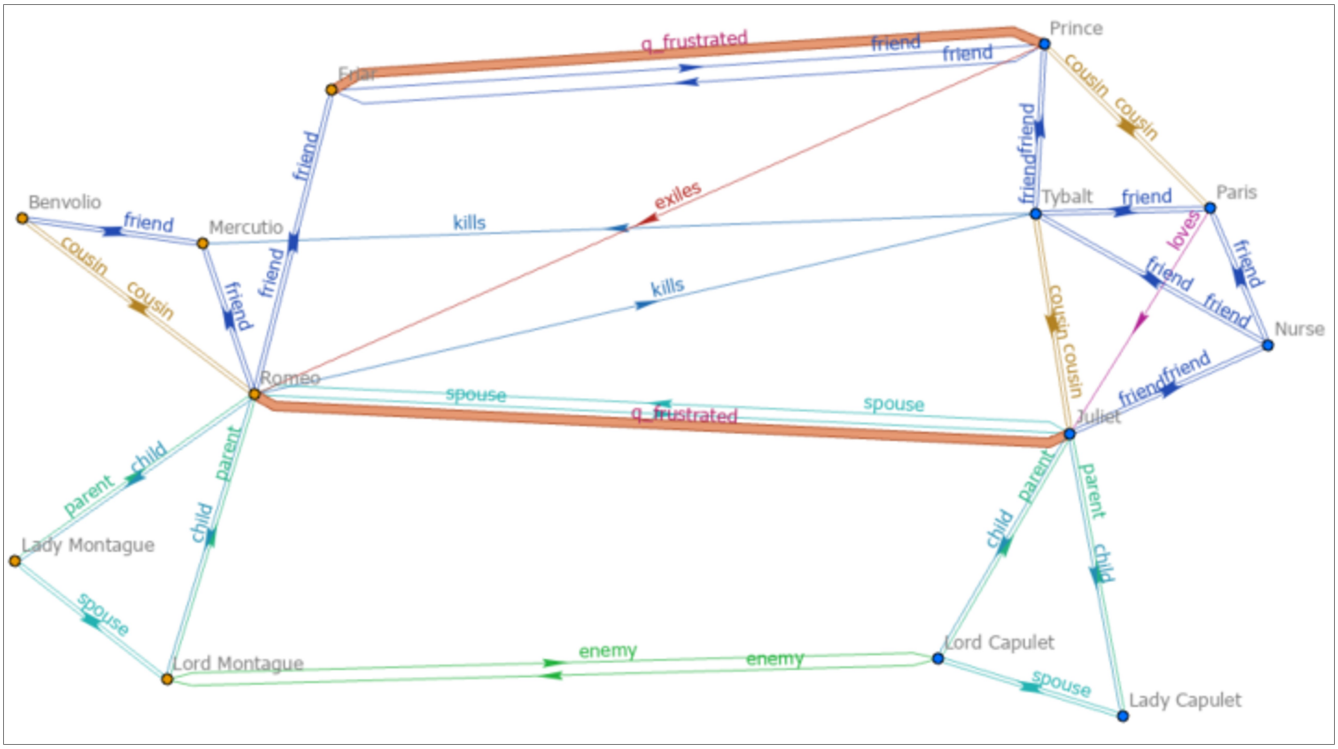
The relationships that break the friendly/hostile rules are considered *frustrated*.



A simple Shakespearean Example

A typical example for this type of problem is Romeo & Juliet. At the beginning of the play, the characters sit in a perfectly balanced graph, all of the Montague and Capulet families have positive relationships within their families, and all relationships between the individuals of the two families are negative. As the story progresses, a frustrated relationship emerges with the title characters. If the relationship between the title characters is updated to reflect their positive interactions and the algorithm is re-run, the relationship is flagged as being frustrated, the two characters are in a friendly relationship when they *should* be hostile. This matters because these frustrated relationships can be a predictor of conflict as they are in this story, but also in real life examples.


Friar
friend
friend
friend
Benvolio
friend
Mercutio
Prince
cousin
cousin
friend
friend
exiles
friend
friend
cousin
cousin
kills
Romeo
kills
Tybalt
friend
Paris
loves
friend
friend
friend
parent
child
parent
child
spouse
spouse
cousin
cousin
Nurse
Lady Montague
friend
friend
spouse
Juliet
Lord Montague
parent
child
parent
child
enemy
enemy
Lord Capulet
spouse
Lady Capulet

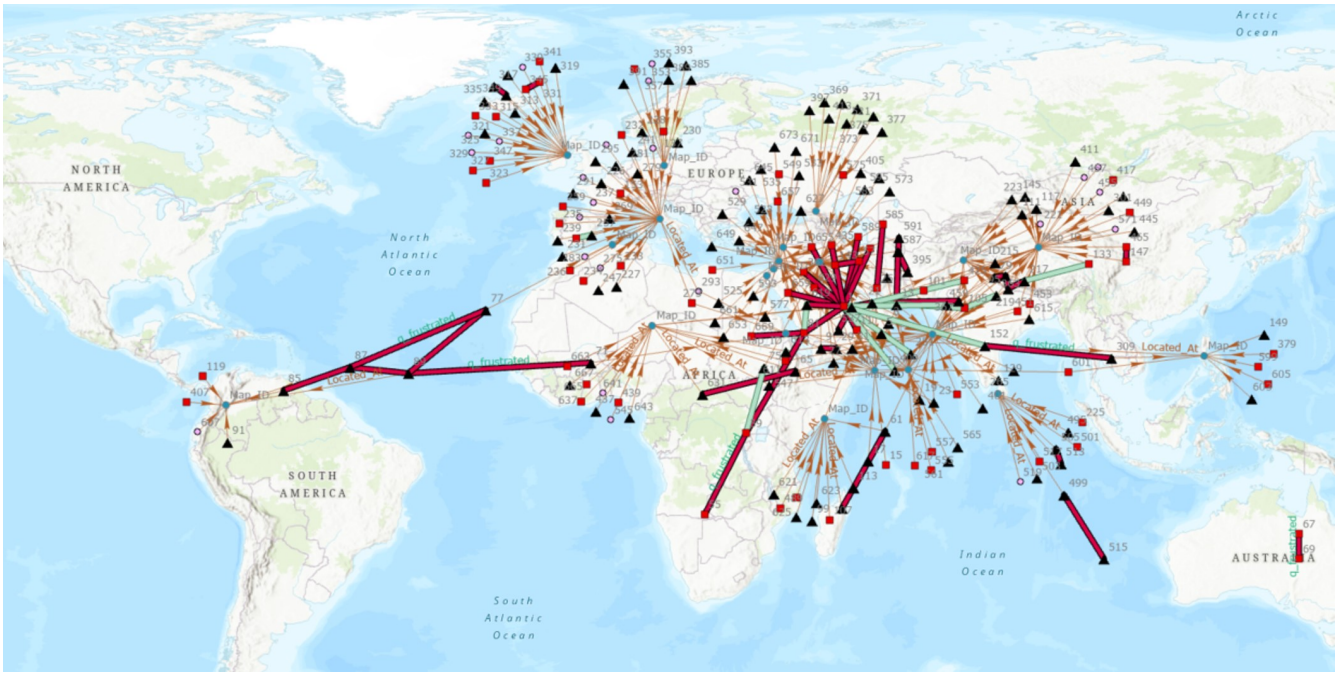


A real world geospatial example

Applying SIP to geospatial use cases requires:

1. A geospatially enabled dataset.
2. A knowledge graph technology that can handle geospatial operators.

An experiment was carried out using some world terrorist incident data, again provided by D-Wave. As with the TSP example, the objective of this piece of work was to integrate AQC and SIP with geospatial technologies to take advantage of geospatial intelligence with AQC to make some observations about patterns found in the data. A note on the parameters for the data is that the dataset was considered as a whole and not split regionally, which is something that could be done in a real world scenario. Additionally, due to the large number of data points, the problem set is too large for the QPU alone and a *hybrid solver* was used instead. Hybrid solvers, as the name suggests, use a combination of classical and quantum technologies to solve larger problems than a pure QPU could do alone. Exactly how the problem is split up is not described in detail, however it does appear to produce timely and accurate results.



In the image above, the red lines show relationships that are hostile, but should be friendly, and the green lines show relationships that are friendly, but should be hostile. Although the grouping created by the AQC are arbitrary, as the dataset has an unconsidered temporal element, it does highlight areas of the world with many frustrated relationships. The Middle East region is particularly challenging with many frustrated relationships that can be a source of conflict.

Spatial dataset sampling

With many organizations choosing to make use of cloud facilities, cost of compute and storage is a consideration for cloud strategy. There are different approaches to this, whether choices are made to keep on-premise infrastructure, often for cost and sovereignty reasons, or to go fully to the cloud, or some hybrid solution where appropriate services are held on-premise and others moved to the cloud. Regardless of the approach, using the cloud requires in a different cost model to on-premise *tin*, as the cloud is a metered service, and on premises compute is a capital expenditure that depreciates over time.

Machine learning and use of Graphical Processing Units (GPU) is a rapid method of building and training machine learning models. ML requires a lot of data to create good models, however, not all data points are critical to the success of the model building process. Appropriate sampling can reduce the amount of data required for machine learning models whilst retaining the explanatory and powerful elements of the model. A reduction in the data required for model building can result in lower cloud costs (as well as compute time and it's implications for environmental concerns).

This type of data sampling requires that the most explanatory data points are kept and the others discarded for processing. In geospatial data, a simple use case is location and difference.

Geospatial sampling selection

Geospatial data has always been big, complex, multidimensional and challenging to manage. Sampling and selection is one method of reducing data, however, calculating similarity between geospatial locations can be approached using a *Haversine function*. This method determines the

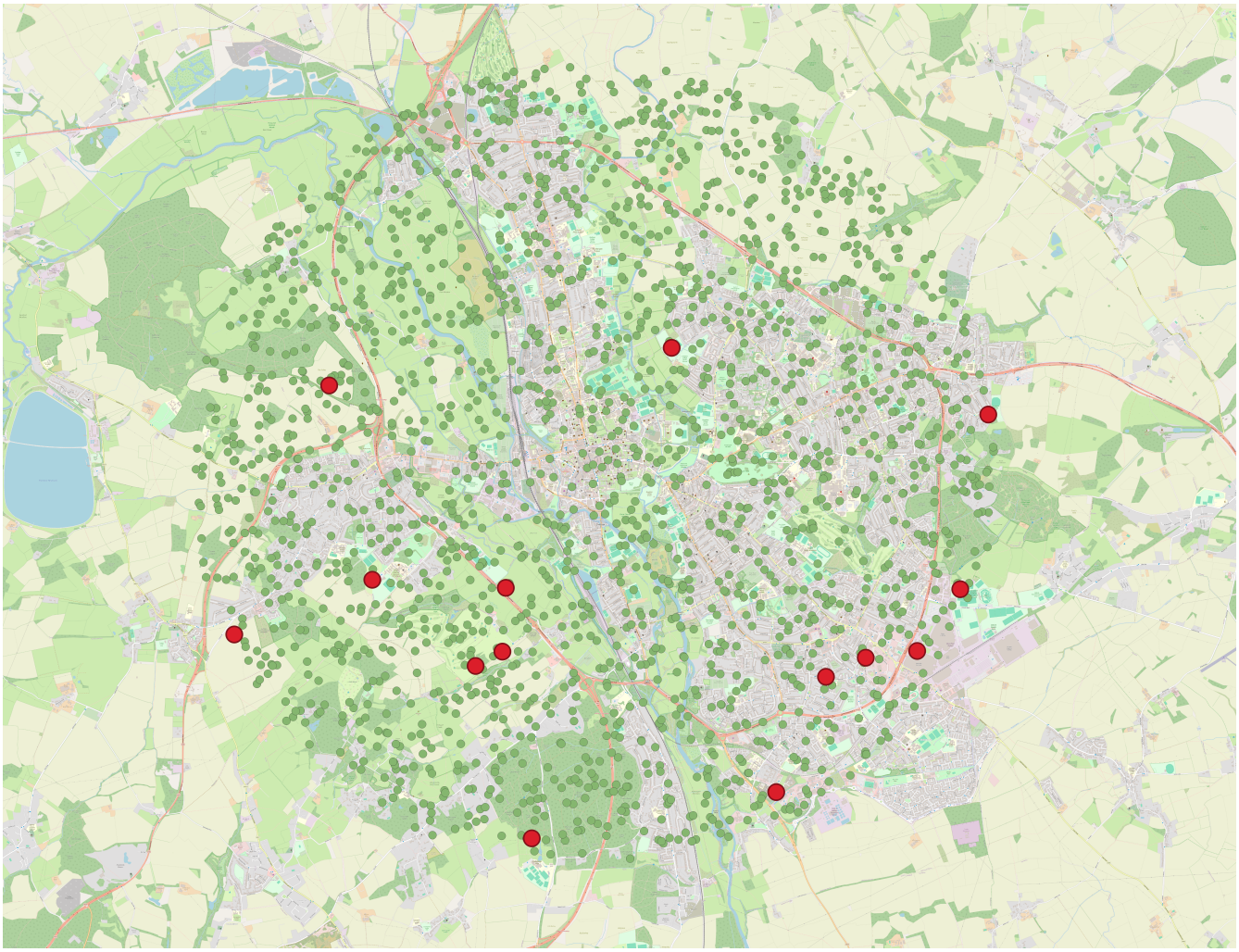
great circle distance between two points given their latitude and longitude. To perform sample selection using AQC, a matrix is created using the Haversine function to compare each position with every other position. Once the matrix is created, which is admittedly a computationally expensive task, QC can be used to down select the dataset to a sample containing the variables that are most *different*.

The QC algorithm works utilizes a *penalty* coefficient and a global subset size constraint to achieve the results. In this example, the global subset size was 15, which means that from the sample of 2000 points, the QC is encouraged to select the top 15 most representative points.

The dataset used is a set of randomly generated points around Oxford in the UK. The purpose of the exercise is to reduce the number of points to a representative sample using a hybrid solver within the D-Wave quantum computer.

Geospatial sampling results

The dataset contained 2000 points and the algorithm was configured to reduce the points to 10%. The results are below. The original data sample is displayed in green and the reduction by quantum algorithm is in red.



Other examples were attempted with different levels of remaining samples, however, there were occasions where the algorithm did not appear to give a representative sample by biasing points in a clustered geographic region. The reasons for this are unknown at time of writing, but more

experimentation is required.

Space example

As optimization can be applied to typical geospatial operations, they can also be applied to any domain where there is a problem with many correct solutions but one being optimal.

Organization, monitoring and controlling satellites whilst taking note of other orbital objects such as space debris contains many optimization problems. Conceptually there are many ways to configure a constellation to achieve certain goals, but there is an optimal solution. Some satellite optimization problems that could be solved using AQC are as follows:

- Coverage Optimization: Ensuring maximum coverage of specific areas on Earth.
- Communication Links: Maximizing the efficiency of communication links between satellites.
- Fuel Efficiency: Minimizing fuel consumption for maneuvers.
- Redundancy and Resilience: Ensuring the constellation can withstand the failure of one or more satellites.

The example described in this Section has a combination of coverage optimization and redundancy and resilience, however, the parameters could be changed and weighted to favor one element over another if *real* data were being used.

Satellite constellation placement optimization

Satellite usage and placement in a constellation is an interesting and geospatially adjacent issue which AQC can help with. Although it is inherently a geospatial issue with respect to observing a patch of the earth, the problem can be simplified into a QUBO as mentioned previously. The role of geospatial technologies and data is to provide intelligence into the input data. The example shown here does not use information about satellite orbits and periods, it simply assumes that a constellation of satellites can observe a location at a given time. Whether a satellite can observe a location is represented as simple binary, 1 for it can observe and 0 it cannot observe the location. Additionally, the temporal element is considered as slices, the matrix provides 5 time slices and the binary describes whether the satellite can view the location. For example, satellite 0 can see the location at time slice 1, but it cannot see the location at time slice 4.

Satellite	Time Slice 1	Time Slice 2	Time Slice 3	Time Slice 4	Time Slice 5
0	1	0	1	0	1
1	0	1	0	1	0
2	1	1	0	1	0

An additional concern with this is to do with cost as there is a cost associated with not observing the location and equally, there is a cost associated with making changes to a satellite. The model seeks to balance the cost of not observing the location at a time with the cost of operating the satellite, it then selects the satellites to use from within the constellation to the monitoring.

In the above example, requiring a coverage of 2, that is, the geographic area should be observed by

at least two satellites at the required time slice given the penalty for *not* observing the location, and the cost of using the satellite. If we run this using the QC, the algorithm reports that satellites 0 and 1 are required. If we up the coverage requirement to 3, then the algorithm reports that all of the satellites are required. After proving the concept, a further experiment was conducted using 25 satellites and 10 time slices. Again, by manipulating the penalties and costs for satellite coverage and usage respectively, solutions were generated that included most, many, some and a single satellite depending on the requirements.

Although this is quite an immature and simplified experiment, the costs, penalties and use of coverage variables have real-world applications that can optimize satellite constellation coverage. This is particularly salient with respect to the new, small satellites such as Starlink being launched as well as cubesats and other low weight, low cost devices being put into orbit. The next step in this experimentation is to use *real* satellite data including costs and penalties to create a constellation and then measure the constellation against the existing constellations.

Potential Standardization Routes

As the OGC is a standards body, understanding how the community can contribute to standardization of quantum calls is discussed in this section. The work described in this paper has shown QC and specifically AQC to have current utility and future potential in optimizing geospatial problems. However, AQC is not a replacement for a technology and is likely to form part of a geospatial workflow rather than replace a geospatial workflow.

Standardization efforts are underway in other standards bodies such as IEEE (https://standards.ieee.org/practices/foundational/quantum-standards-activities/), however, there does not appear to be active initiatives, standards, or progress towards standards in this example.

From an OGC Standards perspective, standardizing calls to an AQC is likely to be a profile or implementation OGC API - Processes. This paper does not attempt to create this profile, but here are some considerations for profiling or standardizing optimization of solutions to NP hard geospatial problems.

- Binary Quadratic Model. The principle of a BQM is to store terms and one or two variables that have a relationship between them. Currently this is held as an array, but metadata could be introduced to describe the data and make it FAIR, or could be setup as a new datatype. A BQM is a generalized form of a QUBO, with a binary of 0 or 1 with the objective of the AQC to minimize the objective function. A BQM can also store an Ising model, which is like a QUBO except the parameters are between -1 and 1.
- Solvers - AQCs use solvers to *run* the problem. There are currently three types of solvers, simulated, QPU and hybrid.
- Peripheral parameters - number of times to run the computation.
- Return types - solutions or raw energy states.

Whether standardization is possible or desirable given the manner that AQCs work will become apparent as the technology matures and becomes adopted over time.

Existing APIs for Quantum machines

At time of writing, there are two accessible APIs for quantum computing, they are:

- D-Wave Leap.
- Amazon Bracket.

These provide a basis and set of requirements for standardization approaches for using quantum solvers as part of an OGC API - Processes (*Processes*) deployment. Processes provides information about Jobs, Processes, Status, and many more relevant calls that could be used to facade quantum computing resources.

Below are two tables that outlines the API calls from both D-Wave and Amazon Bracket. Their corresponding OGC API - Processes call is provided, or where this call might sit.

D-Wave API Call	HTTP Verb	Description	OGC API Reference
/bqm/multipart	POST	Initiate upload of a problem	DRU
/bqm/multipart/<problem_data_id>/part/<part>	PUT	Upload problem data	DRU
/bqm/multipart/<problem_data_id>/combine	POST	Submit a checksum for a problem upload	DRU
/bqm/multipart/<problem_data_id>/status	GET	Status of problem upload	DRU
/problems	POST	Submit problems	DRU
/problems	DELETE	Delete problems	Dismiss
/problems/<problem_id>	DELETE	Delete problem	Dismiss
/problems	GET	Retrieve a list of problems	Job info or process description
/problems/<problem_id>	GET	Retrieve a problem	Job info or Process Description
/problems/<problem_id>/info	GET	Retrieve problem info	Job status info
/problems/<problem_id>/answer	GET	Retrieve answers	callback/execute
/solvers/remote	GET	Retrieve available solvers	Process list
/solvers/remote/<solver_id>	GET	Retrieve solver info	Process info

As mentioned previously, D-Wave is a quantum annealer and bespoke to the D-Wave machines. Amazon Bracket is an Amazon service that abstracts access to multiple quantum computers with

different capabilities and from different vendors.

Amazon Bracket Call	HTTP Verb	Description	OGC API Reference
CancelJob	PUT	Cancels a hybrid job	Dismiss
CancelQuantumTask	PUT	Cancels the specified quantum task	Dismiss
CreateJob	POST	Create a hybrid job	DRU
CreateQuantumTask	POST	Create a quantum job	DRU
GetDevice	GET	Get Device Info	Unknown
GetJob	GET	Get Job info	Job Status Info
GetQuantumTask	GET	Get quantum task info	Job Status Info
ListTaskForResource	GET	Get tags associated with a task	Process Description
SearchDevices	POST	Search for devices using a filter	Unknown
SearchJobs	POST	Search for jobs using a filter	Unknown
SearchQuantumTasks	POST	Search for quantum tasks	Unknown
TagResource	POST	Tag a resource	DRU/unknown
UntagResource	DELETE	Untag a resource	DRU/unknown

This paper does not attempt to extend or create a standard to manage interaction with quantum machines. However, there are elements of OGC API - Processes that could be changed or added to address the quantum use cases.

OGC API - Processes suggestions to support quantum processes

There are several elements that OGC API - Processes needs to support quantum computing. Some suggestions are transferrable to other use cases, such as abstracting over several machines with different capabilities, as with the different quantum machines hosted by Amazon Bracket.

1. The ability to support transaction for injection of new quantum algorithms and associated data is a requirement. Quantum is still in its infancy and therefore, there are many use cases and algorithms that have not been created or thought of.
2. Some APIs, particularly those that aggregate and offer multiple services such as Amazon Bracket require specification of quantum machines. Currently, OGC API - Processes offers a facade on one logical machine, with a set of processes. In quantum, the machines are not generalized, therefore specifying a machine is important.
3. Searching through jobs, processes, and devices should be supported natively in OGC API - Processes.
4. In the D-Wave approach, the datatype for quantum computing is a QUBO or BQM regardless of

the use case or problem domain. Therefore there is a separation of data upload and solver that it is applied to, OGC API - Processes should support this interaction type.

Discussion

The geospatial and space domains contain a multitude of optimization problems that QC can address in its current state with problem sizes increasing in the future. The work described in this discussion paper has provided some examples of where quantum optimization could be advantageous now. Although quantum computers are not powerful and coherent enough to provide a large uplift in processing power and an increase in speed, practical quantum computing is now possible for experimentation and future proofing. In terms of proving a quantum speed up, there are some clear use cases, especially in the logistics domain that have shown promising advantages to use quantum or hybrid solvers. A D-Wave demo involves routing 5 different drivers to satisfy delivery of multiple packages to different locations. This is a very hard problem for classical computers to solve as it involves multiple instantiations of TSP across multiple people. When directly compared to K-Means clustering, the quantum-hybrid algorithm saves around a quarter of the mileage. These types of gains in quantum optimization can make a difference to organizations as they currently stand.

Creation and execution of quantum optimization problems can be reduced to either a BQM or a QUBO. These are useful constructs as having a generic approach to describing problems through generalized calls and abstracted data structures implies that standardization could be a useful exercise for geospatial optimization problems. It is not felt that standardization of calls or data structures related to quantum computing is currently required, as the community is largely experimental. It is also recognized that the OGC are a geospatial community focused on geospatial problems. Although geospatial provides an excellent set of use cases for quantum computing, it is recognized that there are many other communities that will have vested interests in QC and how it is tasked and accessed.

Recommendations

From the background and three use case areas discussed, here are some findings are recommendations:

- Quantum computing has a role to play in solving complex, geospatial optimization problems now and in the future. However, quantum computing is not a replacement for classical approaches and should be used as part of a workflow. Quantum computing should be integrated into more appropriate geospatial workflows.
- There are many geospatial optimization problems can be solved using quantum with the potential to provide an advantage. A list of optimization use cases should be gathered from the geospatial community to focus further research into this emerging technology.
- The immediate uplift in capability is likely to come through using a hybrid solver. The principle of using the hybrid is to task a classical computer with solving the problem that then executes the quantum computer on the user’s behalf. This is a scalable approach to QC that is available now with reported use of millions of entities and hundreds of thousands of variables. The primary interaction with the D-Wave quantum computer should be the hybrid solver, as it

enables quantum computing technologies to solve problems of a useful size potentially providing quantum advantage.

- Quantum annealing cannot be treated as a generalized computing resource as it utilizes quantum tunnelling to find the optimal solution amongst many possible solutions, but cannot be applied to problems such as search, factoring large numbers, or machine learning algorithms (although it maybe used for sample selection).
- The OGC should consider the role standardization could play in quantum computing and quantum technologies more widely. A Domain Working Group (DWG) maybe palatable to support these emerging technologies.
- OGC API - Processes is probably the closet fit to facade QC within the OGC suite of standards. There are several shortcomings with the current standard to support QC that are either already being addressed by extensions including *Deploy, Replace, Undeploy*, but may require abstraction to support different quantum computing approaches.

Conclusions

Quantum computing and related quantum technologies such as cryptography and sensing have varying levels of maturity and have implications for the geospatial domain and standardization. This paper has sought to explore some of these possibilities through experimentation and integration with geospatial technologies to provide results with current quantum machines. The integration path is manageable, and optimization problems should be explored further as the hardware increases capability. Although quantum computers can be executed directly, a near term advantage is found through the use of *hybrid solvers* which use both classical and quantum capabilities to manage larger problems than a QPU alone could manage. == Section header

Annex A: Title

NOTE Place other Annex material in sequential annexes beginning with "A" and leave final two annexes for the Revision History and Bibliography

Annex B: Revision History

Date	Release	Editor	Primary clauses modified	Description
2016-04-28	0.1	G. Editor	all	initial version

Bibliography

NOTE

Example Bibliography (Delete this note).

The TC has approved Springer LNCS as the official document citation type.

> Springer LNCS is widely used in technical and computer science journals and other publications
>
> - For citations in the text please use square brackets and consecutive numbers: [1], [2], [3]
>
> – Actual References:
>
> [n] Journal: Author Surname, A.: Title. Publication Title. Volume number, Issue number, Pages Used (Year Published)
>
> [n] Web: Author Surname, A.: Title, http://Website-Url

- [OGCTB12], *OGC: OGC Testbed 12 Annex B: Architecture* (2015).
- [local-file(bib1, VanZyl2009)]