Telling a Story

Zongzhe Yuan

May 31, 2018

1 Introduction To the Problem

The discussion of our problem originated from a concept that was mentioned in the L11 Algebraic Path Problem class but was not introduced in detail [1]. In L11 class, the concept of reduction was proposed, along with some examples, and was used to solve some problems that we could not solve before. However, in addition to the definition of reduction provided in the lecture, the property of reduction has not been thoroughly discussed. At the same time, although we have found the definition of reduction and practical examples in other papers and thesis before, unfortunately, as in the case of L11, none of them discussed the property of reduction itself in detail and provided a soundness reasoning process. This makes the detailed discussion of reduction a valuable research topic, and becomes to the topic to our project.

In our research, we will not only conduct detailed reasoning on the properties of reduction, but we will also accomplish the following three goals:

- The definition in L11 and the definition of reduction in the previous paper are all defined mathematically, which leads to the representation of reduction become implementation unfriendly. Hence we tried to find an implementation friendly representation equivalent to traditional reduction representation.
- The functionality of classical reduction is limited, and it can not represent some reduction (especially one of the reduction we used in the construction of our path problem). We need a more generalized way to define the reduction, the generalized reduction, and also do reasoning on it.
- Based on the application of reduction to the real problems, we define a class of reduction, predicate reduction on the basis of the previous section generalized reduction. Although predicate reduction is more concrete than generalized reduction (which means that there are some reductions that cannot be represented by predicate reduction), it is a fairly generalized reduction, and we can also use it to define a lot of concrete instance of reduction.

Finally, we will use the predicate reduction defined earlier, combined with the properties of reduction and some other mathematical structures, to define the path problem we encountered in L11 lecture.

2 Basic Definition

Before all the introductions begin, we first introduce a series of mathematical concepts that we will use in our project and will focus on. Let us begin with our problem set S, which is a collection of elements. Here in our project, we force our problem set S to be non-trivial, which means there at least one element inside the problem set.

2.1 Equality

In problem set S, the first thing we need to consider is the equality $=_S$. In the mathematical definition we can think of equality as a particular binary relationship, which is a set of ordered pairs $=_S \in S \times S$. And for equality, we have some properties that need to be discussed.

Congruence:

$$\forall a, b, c, d \in S, a =_{S_1} b \wedge c =_{S_1} d \rightarrow a =_{S_2} c = b =_{S_2} d \tag{1}$$

Reflexive:

$$\forall a \in S, a =_S a \tag{2}$$

Symmetric:

$$\forall a, b \in S, a =_S b \rightarrow b =_S a \tag{3}$$

Transitive:

$$\forall a, b, c \in S, a =_S b \wedge b =_S c \rightarrow a =_S c \tag{4}$$

2.2 Unary Operator

What we need to discuss next is the unary operator acting on S, and our subsequent reduction will be expressed as an unary operator. A unary operator r on S is a function on S, $r : S \rightarrow S$. For the unary operators, we mainly care about the following properties, and these properties are also mainly discussed later when we discuss the properties of reduction.

Congruence:

$$\forall a, b \in S, a =_S b \rightarrow r(a) =_S r(b) \tag{5}$$

Idempotent:

$$\forall a \in S, r(a) =_S r(r(a)) \tag{6}$$

Preserve Id: Given a binary operator $\oplus : S \times S \to S$

$$\exists i \in S, \forall a \in S, a \oplus i =_S a =_S i \oplus a \to r(i) =_S i \tag{7}$$

Preserve Annihilator: Given a binary operator $\oplus : S \times S \to S$

$$\exists a \in S, \forall x \in S, x \oplus a =_S a =_S a \oplus x \to r(a) =_S a \tag{8}$$

Left Invariant: Given a binary operator $\oplus : S \times S \to S$

$$\forall a, b \in S, r(a) \oplus b =_S a \oplus b \tag{9}$$

Right Invariant: Given a binary operator $\oplus : S \times S \to S$

$$\forall a, b \in S, a \oplus r(b) =_S a \oplus b \tag{10}$$

2.3 Binary Operator

The last thing we need to discuss, and also a major focus of this project is the binary operator under problem set S. A binary operator $\oplus$ under S is a function $\oplus : S \times S \to S$. And we will mainly discuss the following properties about the binary operator.

Congruence:

$$\forall a, b, c, d \in S, a =_S b \wedge c =_S d \to a \oplus c =_S b \oplus d \tag{11}$$

Associative:

$$\forall a, b, c \in S, a \oplus (b \oplus c) =_S (a \oplus b) \oplus c \tag{12}$$

Commutative:

$$\forall a, b \in S, a \oplus b =_S b \oplus a \tag{13}$$

Not Commutative:

$$\exists a, b \in S, a \oplus b \neq_S b \oplus a \tag{14}$$

Selective:

$$\forall a, b \in S, a \oplus b =_S a \bigvee a \oplus b =_S b \tag{15}$$

Left Distributive: Given another binary operator $\times : S \times S \to S$

$$\forall a, b, c \in S, a \otimes (b \oplus c) =_S (a \otimes b) \oplus (a \otimes c) \tag{16}$$

Not Left Distributive: Given another binary operator $\times : S \times S \to S$

$$\exists a, b, c \in S, a \otimes (b \oplus c) \neq_S (a \otimes b) \oplus (a \otimes c) \tag{17}$$

Right Distributive: Given another binary operator $\times : S \times S \to S$

$$\forall a, b, c \in S, (a \oplus b) \otimes c =_S (a \otimes c) \oplus (b \otimes c) \tag{18}$$

Not Right Distributive: Given another binary operator $\times : S \times S \to S$

$$\exists a, b, c \in S, (a \oplus b) \otimes c \neq_S (a \otimes c) \oplus (b \otimes c) \tag{19}$$

3 Semiring and Path Problem

Before we going into our real problem, let us introduce a basic definition, called semiring. This section will start with a basic mathematical structure called "semiring". Then we will mention why semiring will be applied to solve the path problem.

3.1 Semiring

In abstract algebra, a semiring is a data structure $(S, \oplus, \otimes, \bar{0}, \bar{1})$ where S is a set (Type) and $\oplus, \otimes$ are two binary operators : $S \times S \to S$.

$(S, \oplus)$ is a commutative semigroup (has associative property) and $(S, \otimes)$ is a semigroup:

$$\forall a, b, c \in S, a \oplus (b \oplus c) = (a \oplus b) \oplus c, a \oplus b = b \oplus a$$

$$\forall a, b, c \in S, a \otimes (b \otimes c) = (a \otimes b) \otimes c$$

$\oplus, \otimes$ are also left and right distributive on S :

$$\forall a, b, c \in S : a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$$

$$\forall a, b, c \in S : (a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$$

$\bar{0}$ is the identity of $\oplus$ and $\bar{1}$ is the identity of $\otimes$:

$$\forall a \in S, a \oplus \bar{0} = a = \bar{0} \oplus a$$

$$\forall a \in S, a \otimes \bar{1} = a = \bar{1} \otimes a$$

Finally, $\bar{0}$ is the annihilator of $\otimes$:

$$\forall a \in S, a \otimes \bar{0} = \bar{0} = \bar{0} \otimes a$$

Some definition will include that $\bar{1}$ is the annihilator of $\oplus$:

$$\forall a \in S, a \oplus \bar{1} = \bar{1} = \bar{1} \oplus a$$

3.2 Semiring Representation of Path Problem

The path problem has always fascinated mathematicians and computer scientists. At the very beginning, programmer and scientists designed algorithms to solve each of the path problem. The most famous path problem is the shortest distance problem and there are several well-known algorithm that can solve such a problem: Dijkstra's algorithm, Bellman–Ford algorithm, A* search algorithm and Floyd–Warshall algorithm. People use different primitive metrics and various complicated algorithms to solve different path problems.

However, such an approach has its obvious shortcomings. Even at some point, designing a new algorithm can "steal" the ideas of the original algorithm, people must design a completely new independent algorithm in the face of each new problem (new metric), and this makes it difficult to have a generic (or framework) approach to solve this type of problem. Even if the path problem has minor changes to the problem, it is difficult for people to solve the new problem by slightly modifying existing algorithms.

Hence, lots of predecessors have found the algebraic approaches to work around this kind of problem. Using the knowledge of abstract algebra, people find that the routing problem (path problem) can be represent using a data structure called "semiring" $(S, \oplus, \otimes, \bar{0}, \bar{1})$[2, 3, 4, 5, 6]. For example, the popular "shortest path problem" can be represented as $(S, min, +, \infty, 0)$[5] and the "maximal capacity path problem" can be represented as $(S, max, min, 0, \infty)$.

For each path problem that represented as a semiring, we can construct a corresponding matrix semiring that represent the concrete problem (the edges and the distances for the shortest path problem for example). Then using matrix multiplication and stability of the closure (the semiring), we can solve the real problem of each concrete path problem. However, the simple matrix approach can only solve the "trivial" path problem. Those complicated problem, for example the widest shortest path problem that constructed from the shortest path problem semiring and the widest path problem (maximal capacity path problem) semiring using lexicograhpic product, can't be solved by this "traditional" theory approach. Some times even the method can find an optimal solution, there is no guarantee to find all optimal solutions using the classical method.

Therefore, people have found a non-classical theory of algebraic path finding method, so that algebras that violated the distribution law can be accepted. This non-classical theory can handle the problem the simple classical theory can't handle, such as the problem that can't be solved by Dijkstra or Bellman-Ford. This kind of method is dedicated to finding the local optimal solution at first, and then the local optimal solution is exactly the same as the global optimal solution by some verification or some addition restriction on the computation. For example, the famous protocol, the routing information protocol which is based on distributed Bellman-Ford algorithms is one of the non-traditional theory. Here provides two examples that if we following the basis of distributed Bellman-Ford algorithm Protocol (For example RIP protocol) that calculate the path from i to destination j by using the knowledge from its immediate, neighbourhood and applying it own path to the neighbour available to the process, we will get the result by using the matrix multiplication.

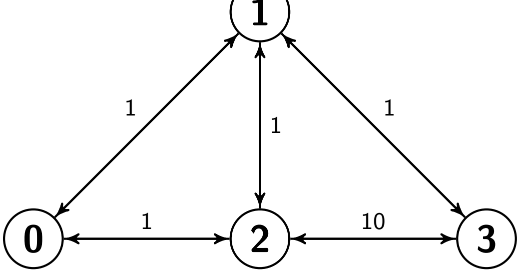

Figure 1: First Example Of RIP solution 1

We will get the following initial path problem adjacency matrix by using the RIP protocol.

$$\begin{bmatrix} \infty & 1 & 1 & \infty \\ 1 & \infty & 1 & 1 \\ 1 & 1 & \infty & 10 \\ \infty & 1 & 10 & \infty \end{bmatrix}$$

After using the RIP protocol for matrix calculations (multiplication), we obtained such an adjacency matrix as the result:

$$\begin{bmatrix} 0 & 1 & 1 & 2 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 2 \\ 2 & 1 & 2 & 0 \end{bmatrix}$$

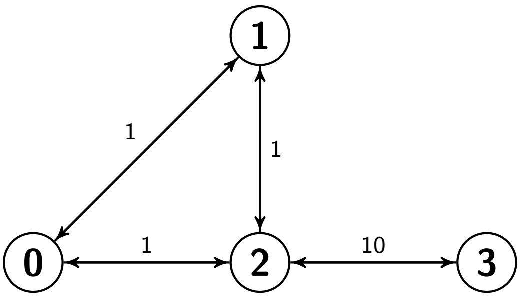

Figure 2: First Example Of RIP solution 2

We will get the following initial path problem adjacency matrix by using the RIP protocol.

$$\begin{bmatrix} \infty & 1 & 1 & \infty \\ 1 & \infty & 1 & \infty \\ 1 & 1 & \infty & 10 \\ \infty & \infty & 10 & \infty \end{bmatrix}$$

After using the RIP protocol for matrix calculations (multiplication), we obtained such an adjacency matrix as the result:

$$\begin{bmatrix} 0 & 1 & 1 & 11 \\ 1 & 0 & 1 & 11 \\ 1 & 1 & 0 & 10 \\ 11 & 11 & 10 & 0 \end{bmatrix}$$

4 Previous Problem in L11

However, even so, RIP will also have a series of problems. For example, when a node does not have edges connected to it, the RIP matrix calculation (without pre-setting the maximum number of calculation steps) will continue infinitely. Even if the maximum number of calculation steps is set in advance, RIP still has some deficiencies in efficiency. This leads to the problem that not all real-world problems can be solved directly with the simple matrix semiring. For example, we may meet the following situation (node 3 is not connected to all other nodes).

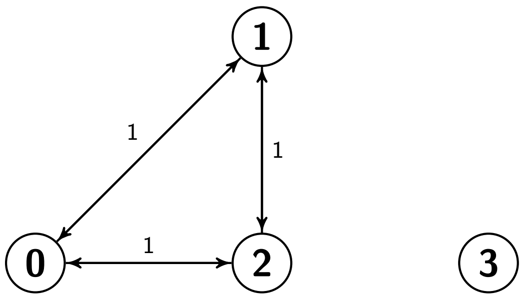

Figure 3: First Example Of RIP solution 3

We will get the following initial path problem matrix by using the RIP protocol.

$$\begin{bmatrix} 0 & 1 & 1 & \infty \\ 1 & 0 & 1 & \infty \\ 1 & 1 & 0 & \infty \\ \infty & \infty & \infty & 0 \end{bmatrix}$$

After using the RIP protocol for n-step matrix calculations (multiplication), we obtained such a matrix:

$$\begin{bmatrix} 0 & 1 & 1 & n+1 \\ 1 & 0 & 1 & n+1 \\ 1 & 1 & 0 & n+1 \\ \infty & \infty & \infty & 0 \end{bmatrix}$$

The result shows that if we do not limit the number of steps in the calculation, eventhough we may get the results of the path we need in the early step (eg path from a to b), but a better solution for the entire matrix may be found after "many: steps (or the calculation will counting to infinity, which is shown in this example).

4.1 Possible Solution

Therefore, in the course L11, instead of using RIP and simple matrix semiring approach, we used another protocol called BGP. We start from the simple $(\mathbb{N}, min, +)$ semiring that calculate the shortest distance, and we use lexicographic product (need reference to the Background/Definition) to construct a new semiring that contains the shortest-path metric and the set of its path.

Here we need to define some new operators/new rules for our semigroup. Assume $(S, \bullet)$ is a semigroup. Let

$$lift(S, \bullet) \equiv (fin(2^S), \hat{\bullet}) \tag{20}$$

where $X\hat{\bullet}Y = \{x \bullet y | x \in X, y \in Y\}$.

Then we can use our $lift$ to construct a bi-semigroup. Assume $(S, \bullet)$ is a semigroup. Let

$$union\_lift(S, \bullet) \equiv (\mathcal{P}(S), \cup, \hat{\bullet}) \tag{21}$$

where $X\hat{\bullet}Y = \{x \bullet y | x \in X, y \in Y\}$, and $X, Y \in \mathcal{P}(S)$, which is the set of finite subsets of S.

Then for a given graph $G = (V, E)$, we define

$$path(E) \equiv union\_lift(E^*, .) \tag{22}$$

where . is the concatenation function of sequence.

Finally we get our "shortest paths with paths" semiring from a given graph $G = (V, E)$:

$$spwp \equiv AddZero(\bar{0}, (\mathbb{N}, min, +) \overrightarrow{\times} path(E)) \tag{23}$$

4.2 Introduce Reduction into Our Problem

Here comes to the problem, when we are using $spwp$ for doing calculations, because there may have loops in our $path(E)$, we need a lot of extra computation (though it will eventually yield correct results) to prove the paths have loops are not the shortest path. In order to eliminate these paths with loops, we introduced a new concept in the L11 course, the reduction.

If $(S, \oplus, \otimes)$ is a semiring and r is a function from S to S, then r is a reduction if $\forall a, b \in S$,

$$r(a) = r(r(a))$$

$$r(a \oplus b) = r(r(a) \oplus b) = r(a \oplus r(b))$$

and

$$r(a \otimes b) = r(r(a) \otimes b) = r(a \otimes r(b))$$

And, if $(S, \oplus, \otimes)$ is a semiring and r is a reduction, then $red_r(S) = (S_r, \oplus_r, \otimes_r)$, where

$$S_r = \{s \in S | r(s) = s\}$$

$$x \oplus_r y = r(x \oplus y)$$
$$x \otimes_r y = r(x \otimes y)$$

After that we went back to our path problem that for a given path p, we say p is elementary if there is no node inside p that is repeated. Then we can define our elementary path using the reduction

$$r(X) = \{p \in X | p \text{ is elementary } \} \tag{24}$$

and

$$epaths(E) = red_r(paths(E)) \tag{25}$$

Therefore, our path problem semiring became to

$$AddZero(\bar{0}, (\mathbb{N}, min, +) \overrightarrow{\times} epath(E)) \tag{26}$$

However, we still encounter the problem that there exists elements in our problem set that has a path distance value but does not have edges in its path. So we need to define the second reduction, which turns all elements that don't satisfy the condition into a single element (the *zero* we added into our semiring).

$$\begin{array}{rcl} r_2(inr(\infty)) & = & inr(\infty) \\ r_2(inl(s, \{\})) & = & inr(\infty) \\ r_2(inl(s, W)) & = & inl(s, W) \end{array} \tag{27}$$

Therefore, our path problem semiring becomes to

$$red_{r_2}(AddZero(\bar{0}, (\mathbb{N}, min, +) \overrightarrow{\times} epath(E))) \tag{28}$$

It is worth mentioning that we did not discuss the properties of reduction in detail in the L11 course. We also did not know whether the original semiring still satisfies the semiring property after the reduction. At the same time, whether some properties of the original operators (such as commutative rules) remains or not after reduction is also a mystery to us.

These are the topics that we need to focus on in our discussion. It is worth mentioning that the first reduction (reduce all pathes that are not elementary) that we defined above does not follow the property of the reduction on the *min* operation (doesn't have left/right invariant property). Imagine we have path $A = \{a, b, c, d\}$ that is longer but does not have loop and $B = \{a, b, b\}$ that is shorter but does have loop. Then $r(A \oplus B) = r(B) = \infty$ because B is shorter but has loop, but $r(A \oplus r(B)) = r(A \oplus \infty) = r(A) = A$ because B has loops but A doesn't.

Therefore, we not only discuss the properties of reduction, but also try to find another way to represent reduction to solve our path problem.

5 Classical Reduction, Example and Reasoning

5.1 Origin of the Concept of Reduction.

In order to better understand reduction and further analyze it in detail, we trace the source to find out if there is any other definition and study of reduction before us. In fact, the earliest definition we could find about reduction came from Ahnont Wongseelashote in 1977[3]. Wongseelashote also encountered the problem similar to ours when studying the path problem in the paper. In fact, our definition of reduction is very similar to Wongseelashote's definition of reduction (the real reason is that L11 refers to the Wongseelashote definition when the definition of reduction was given). However, unfortunately, Wongseelashoth only put forward the concept of reduction in the paper without detailed reasoning and structure proof.

Next we found that Alexander James Telford Gurney also mentioned the concept of reduction in his Ph.D thesis[7]. Gurney's discussion of reduction is also based on the definition by Wongseelashote, and it also does not focus on reduction for detailed structure proof.

5.2 Classical Reduction Definition

Therefore, here we refer to the definition of reduction in L11 (also the definition of Wongseelashote in paper) as the classical reduction, and we give the name of the way in which Wongseelashote represented reduction in paper as the traditional representation of reduction.

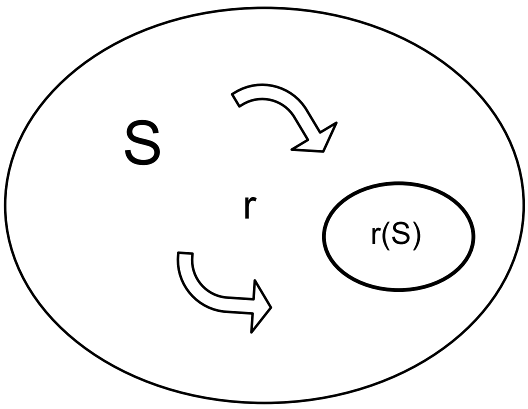

Figure 4: Illustration of Reduction

For a problem set S, the reduced problem set under reduction $r : S \to S$ is represented as

$$r(S) \equiv \{s \in S | r(s) = s\} \tag{29}$$

Then for the problem set S that represented as a semirgroup $(S, =_S, \oplus)$, the reduced problem set will be represented as $(r(S) \equiv \{s \in S | r(s) = s\}, =_S, \oplus_r)$ where

$$\forall a, b \in \{r(S) \equiv \{s \in S | r(s) = s\}, a \oplus_r b \equiv r(a \oplus b) \tag{30}$$

It is worth mentioning that although we use reduction to reduce the problem set from S to $\{r(S) \equiv \{s \in S | r(s) = s\}$, our equality $=_S$ is still the original equality $=_S$, . This can be seen from the figure (5.2), although we currently only focus on a subset of the original problem set S, the equality established on the problem set is still established on this subset. Here as the reduction r need to have the following properties.

Congruence:

$$\forall a, b \in S, a =_S b \to r(a) =_S r(b) \tag{31}$$

Idempotent:

$$r(a) = r(r(a)) \tag{32}$$

Left Invariant:

$$r(a \oplus b) = r(r(a) \oplus b) = r(a \oplus r(b)) \tag{33}$$

Right Invariant:

$$r(a \otimes b) = r(r(a) \otimes b) = r(a \otimes r(b)) \tag{34}$$

5.3 Example on L11 Lecture

To help us understand reduction and study its properties, we found another example of reduction in the course of L11.

The reduction is call $min_{\leq}$ (Martelli's semiring)[8] which means removing all the superset. For a given graph $G = (V, E)$, A cut set $C \in E$ for nodes i and j is a set of edges such there is no path from i to j in the graph $(V, E - C)$. C is minimal if no proper subset of C is a cut set. And Martelli's semiring is such that $A^{(*)}(i, j)$ is the set of all minimal cut sets for i and j. So Martelli's semiring =

$$(S, \oplus, \otimes, \bar{0}, \bar{1})$$

where

$$S = min_{\leq}(2^{2^E})$$

$$X \oplus Y = min_{\leq}(\{U \cup V | U \in X, V \in Y\})$$

$$X \otimes Y = min_{\leq}(X \cup Y)$$

$$\bar{0} = \{\{\}\}$$

$$\bar{1} = \{\}$$

Here we can easily prove that $min_{\leq}$ satisfies our previous definition of reduction. So $min_{\leq}$ is a classical reduction.

5.4 Reasoning on Classical Reduction

6 New Reduction Representation

7 Generalized Reduction

8 Predicate Reduction

9 Path Problem Construction

References

[1] Timothy Griffin. Algebraic path problem, 2017.

[2] B. A. CARRÉ. An Algebra for Network Routing Problems. *IMA Journal of Applied Mathematics*, 7(3):273–294, June 1971.

[3] Ahnont Wongseelashote. Semirings and path spaces. *Discrete Mathematics*, 26(1):55 – 78, 1979.

[4] Seweryn Dynerowicz and Timothy G. Griffin. On the forwarding paths produced by internet routing algorithms. In *Network Protocols (ICNP), 2013 21st IEEE International Conference on*, pages 1–10. IEEE, 2013.

[5] Mehryar Mohri. Semiring frameworks and algorithms for shortest-distance problems. *Journal of Automata, Languages and Combinatorics*, 7(3):321–350, 2002.

[6] Alexander JT Gurney and Timothy G. Griffin. Lexicographic products in metarouting. In *Network Protocols, 2007. ICNP 2007. IEEE International Conference on*, pages 113–122. IEEE, 2007.

[7] Alexander James Telford Gurney. Construction and verification of routing algebras.

[8] Alberto Martelli. A gaussian elimination algorithm for the enumeration of cut sets in a graph. 23(1):58–73.