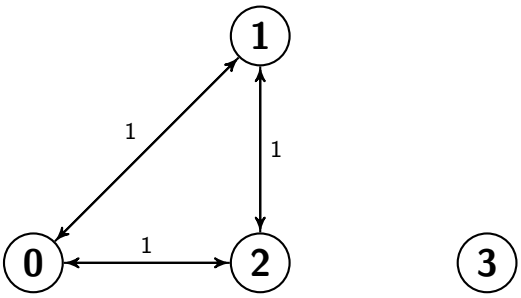
Telling a Story

Zongzhe Yuan

May 26, 2018

1 Initial Approach to the Reduction Problem

The discussion of the reduction problem originated from an extension of a problem that was not discussed in detail in the L11 Algebra Path Problem class.
In the course, we were using the algebraic approaches to represent and calculate the path problem. A path problem will be represented as a semiring and the real case to this problem will be represented and calculated by using a matrix semiring.
However, not all real-world problems can be solved directly with the simple matrix semiring. For example, when we meet the following situation (node 3 is not connected to all other nodes).



We will get the following initial path problem matrix by using the RIP protocol (here need the reference for RIP).

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

After using the RIP protocol for n-step matrix calculations, we have obtained such a matrix.

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

This means that if we do not limit the number of steps in the calculation, 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 the previous example).

Thus, in the course L11, instead of using RIP and simple matrix semiring approach, we used another protocol called BGP (here need the reference for 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.

(these definition could be moved into definition section)
Here we need to define some new operators/new rules for our semigroup:
Assume $(S, \bullet)$ is a semigroup. Let $lift(S, \bullet) \equiv (fin(2^S), \hat{\bullet})$ 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 $union\_lift(S, \bullet) \equiv (\mathcal{P}(S), \cup, \hat{\bullet})$ 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 $path(E) \equiv union\_lift(E^*, .)$ where . is the concatenation function of sequence.
Finally we get our "shortest paths with paths" semiring from a given graph $G = (V, E)$: $spwp \equiv AddZero(\bar{0}, (\mathbb{N}, min, +) \overrightarrow{\times} path(E))$.

Here comes to the problem, when we are using spwp for 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 that the paths that have loops are not the shortest path. Therefore, in order to eliminate these paths with loops, we have introduced a new concept in the L11 course, 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 $r(a \otimes b) = r(r(a) \otimes b) = r(a \otimes r(b))$.
So, if $(S, \oplus, \otimes)$ is a semiring and r is a reduction, then $red_r(S) = (S_r, \oplus_r, \otimes_r)$, where $S_r = \{s \in S | r(s) = s\}$, $x \oplus_r y = r(x \oplus y)$ and $x \otimes_r y = r(x \otimes y)$.
Hence we return to our path problem. Fro 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 $r(X) = \{p \in X | p \text{ is elementary}\}$ and $epaths(E) = red_r(paths(E))$.

Therefore, our path problem semiring becomes to $AddZero(\bar{0}, (\mathbb{N}, min, +) \overrightarrow{\times} epath(E))$.

However, we still encounter the problem that an element in the matrix that has a path distance value but does not have edges in its path. So we need to define a second reduction, which turns all elements that don't satisfy the condition into a single element (the zero we added into our semiring).
$r_2(inr(\infty)) = inr(\infty)$, $r_2(inl(s, \{\})) = inr(\infty)$ and $r_2(inl(s, W)) = inl(s, W)$ where W is not empty. Therefore, our path problem semiring becomes to $red_{r_2}(AddZero(\bar{0}, (\mathbb{N}, min, +) \overrightarrow{\times} epath(E)))$.

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, the properties of some of the original operators (such as commutative rules) remains after reduction or not is also a mystery to us.
These are the topics that we need to focus our discussion on. 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 we defined for reduction on the *min* operation. Imagine we have path $B = \{a, b, c, d\}$ 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 the problem of our path problem.

2 Classical Reduction and Example

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[1]. 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 defining reduction). 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[2]. 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.

Therefore, we can temporarily use the previously mentioned definition of reduction as "classical reduction" and reasoning (because it does not satisfy our elementary path problem and is extraction and implementation un-friendly, so we call it classical).
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)[3] which means remove 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 $S = min_{\leq}(2^{2^E})$, $X \oplus Y = min_{\leq}(\{U \cup V | U \in X, V \in Y\})$, $X \otimes Y = min_{\leq}(X \cup Y)$, $\bar{0} = \{\{\}\}$ and $\bar{1} = \{\}$.
Here we can easily prove that $min_{\leq}$ satisfies our previous definition of reduction. So $min_{\leq}$ is a classical reduction. (I am considering whether or not paste the proof of matrelli's semiring)

3 Generalized Reduction

The previous section mentioned that our definition of classical reduction is implement un-friendly. This is because in the classical reduction, our actual problem set is $S_r = \{s \in S | r(s) = s\}$ with record. Record is difficult to be accurately represented in most situations. Imagine we have two different reductions r_1 and r_2 for S, then the problem set after applying reduction r_1 and r_2 will be $S_r = \{y \in \{x \in S | r_1(x) = x\} | r_2(y) = y\}$. The changing of our problem domain makes all the reasoning and calculation step into trouble. At the same time, since our real problem set has changed from the original S to the current $S_r = \{s \in S | r(s) = s\}$ (suppose we only have on reduction r), all of our operator/relationship domains have changed. If we originally have a binary operator $\oplus : S \to S$, then the binary operator after reduction will become the binary operator $\oplus_r : \{s \in S | r(s) = s\} \to \{s \in S | r(s) = s\}$. And if we apply two/more reductions simultaneously, the entangle of reduction with problem set and relationship/operator makes it almost impossible for us to define the combinator that exists on reduction and solve the problem at all. Taking into account the elementary path problem we previously focused on, in which the reduction does not satisfy the properties of classical reduction, so we need to represent/generalize our reduction in a new way.

How to generalize reduction is exactly what we need to consider. First we take the binary relationship, the equality, as the starting point.
If there is an equality $=$, which is a binary relationship, on the original problem set S, then it is obvious that the equality $=_r$ on the reduced problem set $S_r = \{s \in S | r(s) = s\}$ will no longer to be $=$ any more. If we have $x, y \in \{s \in S | r(s) = s\}$, because we have $r(x) = x$ and $r(y) = y$, then $x =_r y \leftrightarrow r(x) = r(y)$. So we can represent the equality on the reduced set as $\forall x, y \in \{s \in S | r(s) = s\}, x =_r y \equiv r(x) = r(y)$.
For the same problem, we need to generalize the binary operator under the reduction type. If we originally have a binary operator $\oplus : S \to S$, then the binary operator after reduction will become the binary operator $\oplus_r : \{s \in S | r(s) = s\} \to \{s \in S | r(s) = s\}$. Considering that $r(x) = x$ and $r(y) = y$, the binary operator $\oplus_r$ could be represent as $\forall x, y \in \{s \in S | r(s) = s\}, x \oplus_r y \equiv r(r(x) \oplus r(y))$.

After having a new representation method, we need to prove the equivalent of this new method and the classical reduction on representation.
Here we mainly focus on the equivalence of these properties:
Reflexive, Symmetric, Transitive and Congruence for equality(binary relationship) and
Commutative, Associative, Selective, Congruence, Has Identity and Has Annihilator for the binary operator.
So we first define how to represent reduction r on problem set S as a classical reduction.
If we have a semigroup $(S, \oplus)_=$, the reduced semigroup will be $(S_r, \oplus_r)_{=_r}$, where $S_r \equiv \{s \in S | r(s) = s\}$,
$\forall a, b \in \{s \in S | r(s) = s\}, a =_r b \equiv \pi_1 a = \pi_1 b$ (π is the projection function, since the reduced set is represented as a record, we can project the first/second element from the record),and
$\forall a, b \in \{s \in S | r(s) = s\}, a \oplus_r b \equiv (r(\pi_1 x \oplus \pi_1 y), r(r(\pi_1 x \oplus \pi_1 y)) = r(\pi_1 x \oplus$

$\pi_1 y)) \in \{s \in S | r(s) = s\}$.
In this way, we can clearly see that the original representation of classical reduction is very implement unfriendly. Next we try to represent the same problem with our newly defined representation method.
If we have a semigroup $(S, \oplus)_=$, the reduced semigroup will be $(S, \oplus_r)_{=_r}$, where
$\forall a, b \in S, a =_r b \equiv r(a) = r(b)$, and
$\forall a, b \in S, a \oplus_r b \equiv r(r(a) \oplus r(b))$.
Next, we need to show that the properties in the classical representation is isomorphic to the properties in our new representation.
$\forall a \in \{s \in S | r(s) = s\}, a =_r a \equiv \pi_1 a = \pi_1 a \longleftrightarrow \forall a \in S, a =_r a \equiv r(a) = r(a)$ (reflexive)
and more examples, includes equality and binary operator

It is worth mentioning that in our new form of representation, if reduction has a left/right invariant property on the binary operator, then we can conclude that $\forall a, b \in S, a \oplus_r b \equiv r(r(a) \oplus r(b)) = r(a \oplus b)$. But on the other hand, it also shows that our new reduction representation can be used to represent a reduction that does not have an left/right invariant property (as we mentioned earlier in the reduction of the elementary path problem which does not have those properties on *min* operator). This also solves the problem that we previously mentioned that classical reduction cannot be used to define the reduction that we need.

Among all the properties of binary operators (commutative, associative, selective, congruence, has identity and has annihilator), the associative law is the most special one. Take the proof of commutative law as an example:
$\forall a, b \in S, a \oplus_r b \equiv r(r(a) \oplus r(b)) =_r r(r(b) \oplus r(a)) \equiv b \oplus_r a$. And by the definition of $=_r$ we get $r(r(a) \oplus r(b)) =_r r(r(b) \oplus r(a)) \equiv r(r(r(a) \oplus r(b))) = r(r(r(b) \oplus r(a)))$. However, in the proof, we only need to use the commutative law of the original binary operator, which is $\forall a, b \in S, a \oplus b = b \oplus a$, and the property of the congruence of our reduction, which is $\forall a, b \in S, a = b \rightarrow r(a) = r(b)$.
However, the proof of associativity is not the same. We need to proof $\forall a, b, c \in S, (a \oplus_r b) \oplus_r c \equiv r(r(r(a) \oplus r(b)) \oplus r(c)) =_r r(r(a) \oplus r(r(b) \oplus r(c))) \equiv a \oplus_r (b \oplus_r c)$.
In the proof we not only need the associative law of the original binary operator, which is $\forall a, b, c \in S, (a \oplus b) \oplus c = a \oplus (b \oplus c)$, and the property of the congruence of reduction, we also need our reduction to have the nature of left/right invariant.
We've mentioned earlier that the reduction we expect in the elementary path problem does not have these properties on the *min* operator. However, we hope to prove the property of its associative law. So we hope to have a better way to represent associative under reduction, and we call it pseudo-associative: $\forall a, b, c \in S, r(r(r(a) \oplus r(b)) \oplus r(c)) = r(r(a) \oplus r(r(b) \oplus r(c)))$. At the same time we can prove that pseudo-associative and associative under reduction are isomorphic.
So our defined generalize reduction solves both the implement unfriendly and the inability to represent reduction without the left/right invariant properties for classical reduction, while finding another way to prove associative for those reductions that don't have left/right invariant properties.

4 Predicate Reduction

In the previous section we discussed the representation of generalize's reduction, its properties, and that it is isomorphic to traditional reduction. In order to better implement the construction of the path problem we mentioned earlier, we define a type of reduction and perform detailed reasoning on its properties in this section. If we deeply consider our reduction, whether it is to eliminate the path with a loop, or to turn an unqualified element tuple into a specific element –additive identity/multiplicative annihilator, we can easily find that our reduction is in the situation that a predicate is set to reduce the elements that satisfy/not satisfy the condition. Therefore, we call this kind of reduction as predicate reduction.

Initially for our problem set S, we define a predicate P, which is a function from S to *bool* and we need to provide a specific element c (the predicate reduction will reduce all the element that satisfied the predicate to $c \in S$). Then we can define the predicate reduction as $\forall a \in S, r_p(a) = \begin{cases} c & P(a) = true \\ a & otherwise \end{cases}$
We also mentioned in the previous chapters that the reduction we defined represented by the generalize method does not necessarily have the properties of left/right invariant. But as a reduction it must have the properties of the idempontent and congruence, which are $\forall a \in S, r(a) = r(r(a))$ and $\forall a, b \in S, a = b \rightarrow r(a) = r(b)$. So we need to analyse these two properties. For the property of the idempotent, we need to discuss the input element, split the case that the element satisfies the predicate or not. If the element doesn't satisfy the predicate, then $r(a) = a = r(r(a))$. If the element does satisfy the predicate, then $r(a) = c$ and $r(r(a)) = r(c)$, hence we need to prove/provide the property that c must satisfy the predicate $P(c) = true$. For the property of the congruence, we can conclude from $\forall a, b \in S, r_p(a) = \begin{cases} c & P(a) = true \\ a & otherwise \end{cases} = r_p(b) = \begin{cases} c & P(b) = true \\ b & otherwise \end{cases}$ that we need a property $\forall a, b \in S, a = b \rightarrow P(a) = P(b)$, which is the congruence of the predicate P.

5 Construct the Path Problem Semiring

References

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

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

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