Semirings

Advanced Mathematical Structures in Haskell

Pieter Knops Teun Druijf

November 1st, 2018

Outline

Outline

1. Explanation Semirings

2. Semirings in Haskell

What was a Monoid again?

- $(G, \cdot)$

Associativity

$(a \cdot b) \cdot c = a \cdot (b \cdot c)$

Identity element (1)

$1 \cdot a = a = a \cdot 1$

Commutative monoid

- $(G, +)$

Associativity

$(a + b) + c = a + (b + c)$

Identity element (0)

$0 + a = a = a + 0$

Commutative

$a + b = b + a$

Semiring

Theorem

Semiring: *Monoid and Commutative Monoid combined* $(G, +, \cdot)$

Semiring $(G, +, \cdot)$

- $(G, +, \cdot)$

Commutative Monoid

$(a + b) + c = a + (b + c)$
$a + b = b + a$
$0 + a = a = a + 0$

Monoid

$(a \cdot b) \cdot c = a \cdot (b \cdot c)$
$1 \cdot a = a = a \cdot 1$

Extra properties

$0 \cdot a = 0 = a \cdot 0$
$a \cdot (b + c) = a \cdot b + a \cdot c$
$(a + b) \cdot c = a \cdot c + b \cdot c$

Star semiring

Extra operator $*$

$a* = 1 + a * \cdot a = 1 + a \cdot a*$

Example of Semiring

Boolean Semiring $(G, +, \cdot)$

Where $G = \{T, F\}$ with $T =$True, $F =$False.
$+ := ||$
$\cdot := \&\&$

We have our first Semiring

- $(\{True, False\}, ||, \&\&)$

Commutative Monoid

$(a||b)||c = a||(b||c)$
$a||b = b||a$
$F||a = a = a||F$

Monoid

$(a\&\&b)\&\&c = a\&\&(b\&\&c)$
$T\&\&a = a = a\&\&T$

Extra properties

$F\&\&a = F = a\&\&F$
$a\&\&(b||c) = a\&\&b||a\&\&c$
$(a||b)\&\&c = a\&\&c||b\&\&c$

Define the star

- $a* = T||(a\&\&a*) = T||(a * \&\&a)$

Define the star

- $a* = T||(a\&\&a*) = T||(a * \&\&a)$
- So if we define $a* = T$ fo all a in the semiring, it is closed!

Real numbers as Semiring

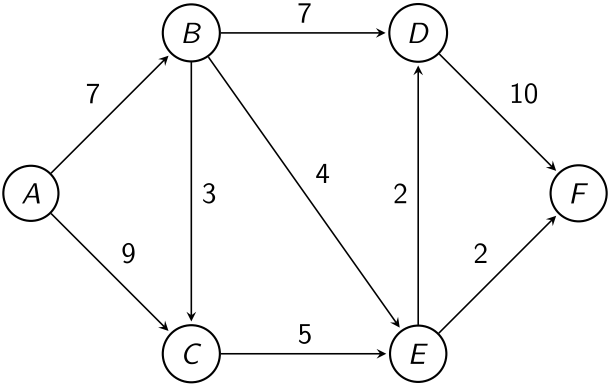
- $(\mathbb{R}, +, \cdot)$
- $a* = \frac{1}{1-a}$
- Extra element $1* = \infty$

Outline

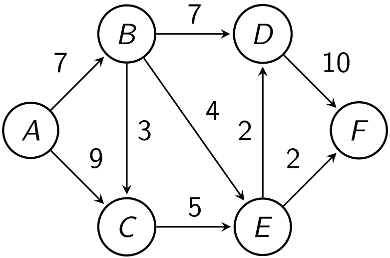
Example

- Riverdelta, each path has a length.
- Goal: Find shortest path from A to F

Our directed graph



Translation to matrix



$$\begin{pmatrix} x & 7 & 9 & x & x & x \\ x & x & x & 7 & 4 & x \\ x & 3 & x & x & 5 & x \\ x & x & x & x & x & 10 \\ x & x & x & 2 & x & 2 \\ x & x & x & x & x & x \end{pmatrix}$$

How can we use Semirings in Haskell to achieve our goals?

- With Haskell we can compute the closure of our matrix:

Closure of Matrix M

$M* = I + M \cdot M* = I + M + M^2 + M^3 + ...$

- We need to define $+$ and $\cdot$ for matrices:

Usual matrix addition and matrix multiplication for n x n-matrix

$(A + B)_{ij} = A_{ij} + B_{ij}$
$(A \cdot B)_{ij} = \sum_{k=1}^{n} A_{ik} \cdot B_{kj}$

ShortestPath instance

```
data ShortestPath n = Path Int [(n,n)]
                    | NoPath
deriving (Show)
```

Semiring instance for ShortestPath

```
instance Ord n => Semiring (ShortestPath n) where
zero = NoPath
one = Path 0 []
closure x = one

x @+ NoPath = x
NoPath @+ x = x
Path a p @+ Path a' p' | a < a'             = Path a  p
                       | a == a'&& p < p'   = Path a  p
                       | otherwise          = Path a' p'

x @. NoPath = NoPath
NoPath @. x = NoPath
Path a p @. Path a' p' = Path (a + a') (p ++ p')
```

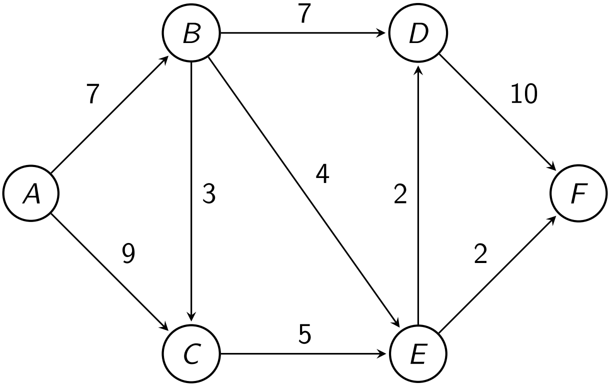
Matrix Translation to semiring instance

```
let presEx = Matrix
[[NoPath, Path 7 [("A","B")], Path 9 [("A","C")],
 NoPath, NoPath, NoPath],
[NoPath, NoPath, NoPath,
 Path 7 [("B","D")], Path 4 [("B","E")], NoPath],
[NoPath, NoPath, Path 3 [("C","B")],
 NoPath, Path 5 [("C","E")], NoPath],
[NoPath, NoPath, NoPath ,
 NoPath,  NoPath, Path 10 [("D","F")]],
[NoPath, NoPath, NoPath,
 Path 2 [("E","D")], NoPath, Path 2 [("E","F")]],
[NoPath, NoPath, NoPath,
 NoPath, NoPath, NoPath]]
```

Output of closure operation

```
$ closure presEx
Matrix [[Path 0 [],Path 7 [("A","B")],
Path 9 [("A","C")],
Path 13 [("A","B"),("B","E"),("E","D")],
Path 11 [("A","B"),("B","E")],
Path 13 [("A","B"),("B","E"),("E","F")]],
...
,[NoPath,NoPath,NoPath,NoPath,NoPath,Path 0 []]]
```

Our directed graph



Q & A

For the complete listing of our Haskell program, we would like to refer you to our document.