Naiad

- Composable data-parallel computation
  - Iterative
    - i.e. arbitrary dataflow graph
  - Incremental
    - i.e. allows fast recomputation after updates

- “Differential Dataflow”
Fixed-point Iterations

Input batches

{A,B}
{A,B,C}
{A,B}
{A}
{A}
{A}
{A}
{A}
{A}
{A}
IDDEN COMPONENTS

public static Collection<IntPair, T> PropagateMin<T>(this Collection<IntPair, T> labels, Collection<IntPair, T> edges)
where T : Lattice<T>
{
    // the Join and Min both use overloads allowing for value selectors.
    return labels.Where(x => false)
    .FixedPoint(x => x.Join(edges.ExtendTime(), n => n.s, e => e.s, (n, e) => new IntPair(e.t, n.t))
    .Concat(labels.ExtendTime())
    .Min(n => n.s, n => n.t),
    n => n.s);
}

public static Collection<IntPair, T> PropagateMinPrioritized<T>(this Collection<IntPair, T> labels, Collection<IntPair, T> edges)
where T : Lattice<T>
{
    return labels.Prioritize(node => Convert.ToInt32(Math.Log(1 + node.t)),
    1 => 1.PropagateMin(edges.Prioritize()));
}

public class ConnectedComponents : Example
Connected components 100000 100000
Logging initialized to console
00:00:00.1187506, Warning: DEBUG build. Not for performance measurements.
00:00:00.1189865, Warning: Using fewer threads than available processors (use -t to set number of threads).
00:00:00.1190933, Initializing 1 thread
Running connected components on a random graph (100000 nodes, 1000000 edges)
For each size, the number of components of that size (may take a moment):
00:00:15.4639558, Epoch changed from 0 to 1
Time to process: 00:00:15.2525559
[ [100000, 1], 1 ]

Next: sequentially removing 10 random edges (press [enter] to start):

Deleting edge: (72624, 81732)
00:00:44.0996193, Epoch changed from 1 to 2
Time to process: 00:00:44.0040397

Deleting edge: (76022, 55816)
00:00:44.1006162, Epoch changed from 2 to 3
Time to process: 00:00:44.0006315