The Milk Problem

Milk matters

The problem asks for a way to divide 8 liters of milk evenly using only three jars of 8L, 5L, and 3L capacity respectively. Of course, the solution is quite obvious, so we go on to ask what volumes of milk we can obtain using those jars, and perhaps using other jars with a different amount of milk.

Solution

Given the jar sizes and the starting condition, we first use recursion to find the possible states of amount of milk in each jar one can achieve. (note: I first wrote this in Python, and I found it difficult to adapt the code to Mathematica nicely, so I definitely missed some easier way to write this).

```
milknodes[moves_List, startstate_List, node_List: {}] := Module[{},
nodes = Union[node, {startstate}];
n = Length[startstate];
Do[Do[newstate = startstate;
    If[i != j && startstate[[i]] > 0 && moves[[j]] > startstate[[j]],
newstate[[i]] = Max[startstate[[i]] + startstate[[j]] - moves[[j]], 0];
     newstate[[j]] = Min[startstate[[i]] + startstate[[j]], moves[[j]]]];
If[! MemberQ[nodes, newstate ], nodes = milknodes[moves, newstate, nodes ]
 ]], {j, n}], {i, n}];
(*recursive step--find nodes connected to
   the new node we have that is not already in the nodes list*)
Return[nodes];
]
```

Then we have the possible states for the milk problem given the initial state, each will be represented as a node on our graph.

```
milknodes[{8, 5, 3}, {8, 0, 0}]
```

```
{{0, 5, 3}, {1, 4, 3}, {1, 5, 2}, {2, 3, 3}, {2, 5, 1}, {3, 2, 3}, {3, 5, 0}, {4, 1, 3},
 {4, 4, 0}, {5, 0, 3}, {5, 3, 0}, {6, 0, 2}, {6, 2, 0}, {7, 0, 1}, {7, 1, 0}, {8, 0, 0}}
```

We can then record the possible routes to go from one state to another to form an adjacency matrix.

```
milkedges[moves_List, startstate_List] := Module[{},
  nodes = milknodes[moves, startstate];
  m = Length[nodes];
  edges = ConstantArray[0, {m, m}];
  Do[curstate = nodes[[k]];
   Do[Do[newstate = curstate;
     If[i != j && curstate[[i]] > 0 && moves[[j]] > curstate[[j]],
      newstate[[i]] = Max[curstate[[i]] + curstate[[j]] - moves[[j]], 0];
      newstate[[j]] = Min[curstate[[i]] + curstate[[j]], moves[[j]]];
      newpos = Position[nodes, newstate] // Last;
      edges[[k, newpos]] = 1;], {j, n}], {i, n}], {k, m}];
  Return[edges];
 ]
```

The adjacency matrix for the 8 - 5 - 3 case looks like :

```
milkedges[{8, 5, 3}, {8, 0, 0}]
```

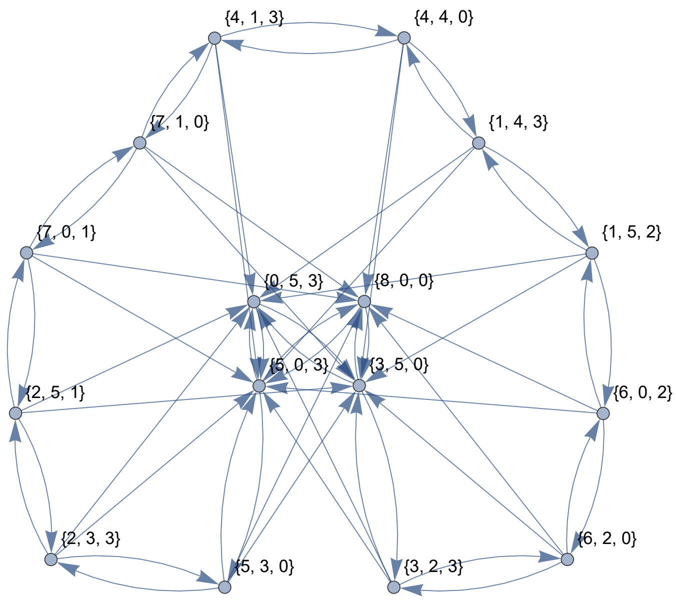
```
{{0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0}, {1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0},
 {1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0}, {1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0},
 {1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0}, {1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0},
 {1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1}, {1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0},
 {0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1}, {1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1},
 {0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1}, {0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1},
 {0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1}, {0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1},
 {0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0}}
```

We can plot the adjacency graph from the matrix,

```
milkgraph[moves_List, startstate_List] := AdjacencyGraph[milkedges[moves, startstate],
  VertexLabels → Table[nodes = milknodes[moves, startstate];
    i → ToString[nodes[[i]]], {i, Length[nodes]}]]
```

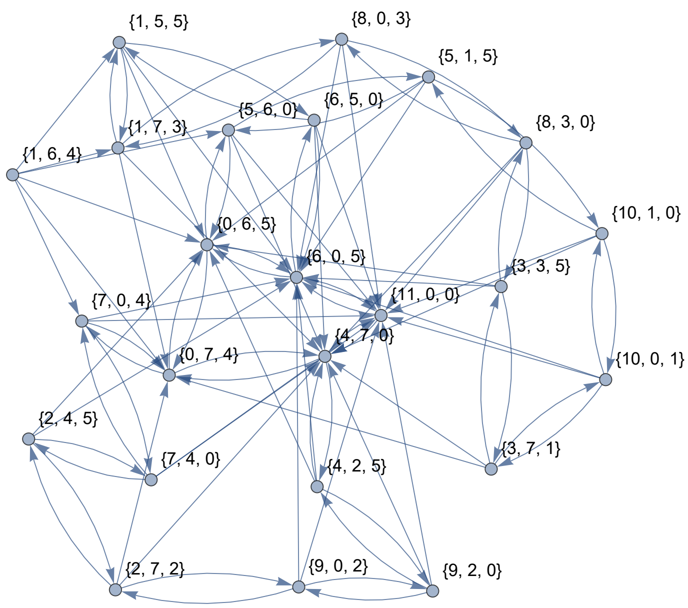
That is, for the 8 - 5 - 3 case, the states graph is :

```
milkgraph[{8, 5, 3}, {8, 0, 0}]
```



This implementation applies to any number of jars and starting state. For example, if we start with jars of 11, 7, and 5 liters, with 1 L, 6 L, and 4 L milk inside respectively, we would have the (much messier) state graph :

```
milkgraph[{11, 7, 5}, {1, 6, 4}]
```



Spillage

Suppose we are allowed to shamelessly Mathematica, I mean waste milk, we have a bit more options

as to where the milk will go. Implementing this is simple-- we are essentially adding another jar of infinite capacity and cannot be filled from.

```
milknodesspill[moves_List, startstate_List, node_List: {} ] := Module[{},
  nodes = Union[node, {startstate}];
  n = Length[startstate];
  Do[Do[newstate = startstate;
    If[i != j && startstate[[i]] > 0 && moves[[j]] > startstate[[j]],
     newstate[[i]] = Max[startstate[[i]] + startstate[[j]] - moves[[j]], 0];
     newstate[[j]] = Min[startstate[[i]] + startstate[[j]], moves[[j]]];
     If[! MemberQ[nodes, newstate ], nodes = milknodesspill[moves, newstate, nodes ]
      ]], {j, n}], {i, n - 1}];
  Return[nodes];
 ]
```

That is,

```
milknodesspill[{8, 5, 3, Infinity}, {8, 0, 0, 0}]
```

```
{{0, 0, 0, 8}, {0, 0, 1, 7}, {0, 0, 2, 6}, {0, 0, 3, 5}, {0, 1, 0, 7}, {0, 1, 1, 6}, {0, 1, 2, 5},
 {0, 1, 3, 4}, {0, 2, 0, 6}, {0, 2, 1, 5}, {0, 2, 2, 4}, {0, 2, 3, 3}, {0, 3, 0, 5}, {0, 3, 1, 4},
 {0, 3, 2, 3}, {0, 3, 3, 2}, {0, 4, 0, 4}, {0, 4, 1, 3}, {0, 4, 2, 2}, {0, 4, 3, 1}, {0, 5, 0, 3},
 {0, 5, 1, 2}, {0, 5, 2, 1}, {0, 5, 3, 0}, {1, 0, 0, 7}, {1, 0, 1, 6}, {1, 0, 2, 5}, {1, 0, 3, 4},
 {1, 1, 0, 6}, {1, 1, 3, 3}, {1, 2, 0, 5}, {1, 2, 3, 2}, {1, 3, 0, 4}, {1, 3, 3, 1},
 {1, 4, 0, 3}, {1, 4, 3, 0}, {1, 5, 0, 2}, {1, 5, 1, 1}, {1, 5, 2, 0}, {2, 0, 0, 6},
 {2, 0, 1, 5}, {2, 0, 2, 4}, {2, 0, 3, 3}, {2, 1, 0, 5}, {2, 1, 3, 2}, {2, 2, 0, 4},
 {2, 2, 3, 1}, {2, 3, 0, 3}, {2, 3, 3, 0}, {2, 4, 0, 2}, {2, 5, 0, 1}, {2, 5, 1, 0},
 {3, 0, 0, 5}, {3, 0, 1, 4}, {3, 0, 2, 3}, {3, 0, 3, 2}, {3, 1, 0, 4}, {3, 1, 3, 1},
 {3, 2, 0, 3}, {3, 2, 3, 0}, {3, 3, 0, 2}, {3, 4, 0, 1}, {3, 5, 0, 0}, {4, 0, 0, 4},
 {4, 0, 1, 3}, {4, 0, 2, 2}, {4, 0, 3, 1}, {4, 1, 0, 3}, {4, 1, 3, 0}, {4, 2, 0, 2},
 {4, 3, 0, 1}, {4, 4, 0, 0}, {5, 0, 0, 3}, {5, 0, 1, 2}, {5, 0, 2, 1}, {5, 0, 3, 0},
 {5, 1, 0, 2}, {5, 2, 0, 1}, {5, 3, 0, 0}, {6, 0, 0, 2}, {6, 0, 1, 1}, {6, 0, 2, 0},
 {6, 1, 0, 1}, {6, 2, 0, 0}, {7, 0, 0, 1}, {7, 0, 1, 0}, {7, 1, 0, 0}, {8, 0, 0, 0}}
```

We have a lot more possible cases in this scenario due to spillage.

```
milkedgesspill[moves_List, startstate_List] := Module[{},
  nodes = milknodesspill[moves, startstate];
  m = Length[nodes];
  edges = ConstantArray[0, {m, m}];
  Do[curstate = nodes[[k]];
   Do[Do[newstate = curstate;
     If[i != j && curstate[[i]] > 0 && moves[[j]] > curstate[[j]],
      newstate[[i]] = Max[curstate[[i]] + curstate[[j]] - moves[[j]], 0];
      newstate[[j]] = Min[curstate[[i]] + curstate[[j]], moves[[j]]];
      newpos = Position[nodes, newstate] // Last;
      edges[[k, newpos]] = 1;], {j, n}], {i, n - 1}], {k, m}];
  Return[edges];
  ]
milkgraphspill[moves_List, startstate_List] :=
 AdjacencyGraph[milkedgesspill[moves, startstate],
  VertexLabels → Table[nodes = milknodesspill[moves, startstate];
    i → ToString[nodes[[i]]], {i, Length[nodes]}]]
```

And so we have a much larger graph.

```
milkgraphspill[{8, 5, 3, Infinity}, {8, 0, 0, 0}]
```