February 20, 2020

Edge relaxation/conditional hooking: Let p be a vector of ints and A be a matrix of EdgeExt (key, weight, parent).
Analgous to $p["i"] += A["i"] * p["j"]$ for hooking.

```
Algorithm:
  while $p_prev != p$:
    $p_prev = p$
    $p["i"] += A["i"] * p["j"]$

    while $s_prev != p$
      $s_prev = p$
      $p[p[i]] = s_prev$
```

Note $+=$ ensures hooking on roots.

For current mst implementation, $p["i"] += A["i"] * p["j"]$ is computed with the following routine:

```
Edge Relaxation:
  auto q = new Vector<EdgeExt>(n, p->is_sparse, *world, MIN_EDGE);
  (*q)["i"] = Function<int,EdgeExt>([](int p){ return EdgeExt(INT_MAX, INT_MAX,
  Bivar_Function<EdgeExt,int,EdgeExt> fmv([](EdgeExt e, int p){ return EdgeExt(
  fmv.intersect_only=true;
  (*q)["i"] = fmv((*A)["ij"], (*p)["j"]);
  (*p)["i"] += Function<EdgeExt,int>([](EdgeExt e){ return e.parent; })((*q)["i
```

For all i, this routine finds computes $minedge = \min_j A["ij"]$ and sets $p[i] = minedge.parent$. However, our current implementation fails to perform any hooking after the initial step. The nodes are "happy" with their miniumum edge and do not want to take any other edge.

Potential fixes: "zeroing out" entries in A to signify that the corresponding edge has already been taken, project matrix to $\#(stars)x\#(stars)$, or enlarge EdgeExt to have an additional boolean parameter *taken*.

We also must consider how to track mst, as p will only contain information about the current forest (ie not where edges originally came from/were going to).