Quick Guide to the Transshipment Problem

The Big Picture

The transshipment simplex algorithm has two phases:

- **Phase I**: Find *any* feasible solution (or prove none exists)
- **Phase II**: Improve that solution until it's optimal

Both phases work with **tree solutions** — the key insight is that basic solutions correspond to spanning trees.

Phase I: Finding a Feasible Solution

1. **Build the auxiliary network**:
   - Pick any source node (say node 1)
   - Add artificial arcs from node 1 to all sinks (demand nodes)
   - Add artificial arcs from all other sources to node 1
   - Set cost = 0 for original arcs, cost = 1 for artificial arcs
2. **Build initial tree**: Use only the artificial arcs connecting to your chosen source
3. **Compute primal solution** x: Work from leaves inward — at each leaf, the flow on the edge equals the supply/demand at that node
4. **Compute dual solution** y: Set $y_1 = 0$, then propagate using $y_j = y_i + c_{ij}$ along tree edges
5. **Run Phase II on auxiliary problem** until optimal
6. **Check result**: If optimal cost = 0, you have a feasible tree for the original problem. If cost > 0, original problem is infeasible.

Phase II: Improving to Optimality

Repeat until no violated constraints:

1. **Compute dual solution** y **from the tree**:
   - Pick any node (usually node 1) and set $y_1 = 0$
   - Traverse the tree, for each edge:
     - If edge goes $(i \to j)$ and you know y_i: set $y_j = y_i + c_{ij}$
     - If edge goes $(j \to i)$ and you know y_i: set $y_j = y_i - c_{ji}$
   - Continue until all nodes have y values
2. **Check dual constraints**: For each non-tree arc (i, j), check if $y_j - y_i \leq c_{ij}$
3. **If violated**: Add arc (i, j) to tree $\Rightarrow$ creates a cycle C
4. **Split cycle**: C_+ = arcs with same orientation as (i, j), C_- = opposite orientation
5. **Compute** Δ: $\Delta = \min\{x_{kl} : (k, l) \in C_-\}$
6. **Update flows**: Add Δ to arcs in C_+, subtract Δ from arcs in C_-
7. **Remove arc**: The arc where minimum was achieved leaves the tree
8. **Go back to step 1** (recompute y with new tree)

Stopping condition: When all dual constraints $y_j - y_i \leq c_{ij}$ are satisfied $\Rightarrow$ optimal!

Intuition

- y_i represents the "price" at node i
- Violated constraint means you can buy at i and sell at j for a profit
- So there's room to improve the solution

Practical Tips

- Always keep the original graph visible — don't erase it
- Check dual constraints systematically (lexicographic order: $(1,2), (1,3), \ldots, (2,3), \ldots$)
- Only check arcs NOT in the current tree (tree arcs satisfy constraints by construction)
- Violated dual constraint = room for improvement