What does the program do?

Inputs:

-Transplant Center, DR#1, DR#2, UA List, Blood type

Outputs (for 0, 1, 2 DR mismatch):

-(historical) number of new potential donors in month/year

-(historical) number of WL patients that may compete for this donor

What data is available?

Supply dataset:

-Supplier (organ) ID, Transplant Center, Offer date, DR#1, DR#2, Blood type

Demand dataset:

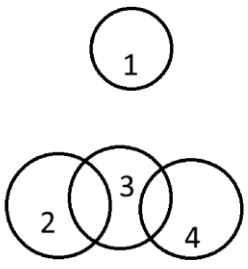
-Consumer (patient) ID, Transplant Center, WL start date, WL end date, DR#1, DR#2, UA, Blood type

Model:

Note on limitations:

-In this model I completely ignore the role of dates in all of this. I don't know the actual methods of the paper so I simplified to the coding task itself.

Center locations



I generate the following crosswalk on distances. Where I have 5 miles (connected) vs 500 miles (unconnected). If there are M centers this table is M^2 in sized. Which should be doable. Otherwise just need to list pairwise connections and use difference coding.

Center 1	Center 2	Distance
1	1	5
1	2	500
1	3	500
1	4	500
2	1	500

2	2	5
2	3	5
2	4	500
3	1	500
3	2	5
3	3	5
3	4	5
4	1	500
4	2	500
4	3	5
4	4	5

Match criteria: pairing between (organ i) and patient (j) when

1. DRs(i)=DRs(j) [this takes on values 0 1 2]
2. Blood type(i) = Blood type(j)
3. Antigens(i) != UAs(j)
4. D(Center (i), Center (j)) < 250. Distance is gathered beforehand in distance lookup table (see above)
5. I ignore any timing details about dates

Call a match U(i,j)(k)=1. Let U(j,i)(k)=0. Where k refers to 0, 1, 2 mismatches.

Time to derive outputs

Supply:

$$\#(\text{Supply for patient } j_*)(\text{k mismatches}) = \sum_{\text{i}} U(i, j_*)(k)$$

Again – I'm not doing anything about the "date" of these events.

1. User specifies Transplant Center and Organ List: $\{j_*\}$ ← user input
2. Find list $\{i_1, \dots, i_I\}$ of organs such that $U(i, j_*)(k) = 1$.
3. Find $\sum_{\text{i}} U(i, j_*)(k)$. Then normalize it to years / months in some way

   (I just report the total sum)

Demand:

$$\#(\text{Demand for organ } i)(\text{k mismatches}) = \sum_{\forall j} U(i, j)(k?)$$

Not clear if demand should be subject to the k mismatches criteria.

Not clear if we have the UAs for those on the WL.

1. User specifies Transplant Center and Organ List: $\{j_*\}$ ← user input
2. Find list $\{i_1, \dots, i_I\}$ of organs such that $U(i, j)(k) = 1$.
3. For each $i \in \{i_1, .., i_I\}$ find a list of interested recipients $\{j_1, \dots, j_J\}_i$ such that $U(i, j)(k?) = 1$
4. Insert some step about double counting say one patient j being interested in 2 organs.
5. Perform some transformation based on timeframe
6. Combine the sums for each organ in patient j_* organ list.

$$\#(\text{Demand for patient } j_*'\text{s potential organs } \{i_1, \dots, i_I\})(\text{k mismatches})$$

$$= \sum_{\forall i \; s.t \; U(i,j_*)(k)=1} \sum_{\forall j} U(i,j)(k?)$$

Coding Structure:

Steps:

1. Simulate Data
2. Create fixed crosswalks
    a. Transplant center distances
    b. {Supplier i, Consumer j}(k) – that is 3 different lists
3. Create JS script that can find the list $\{i_1, \dots, i_I\}$ based on user input $\{j_*\}$
4. Create website that takes user input and outputs details based on above methodology

Simulate Data:

Supply: N=40,000

Demand: N=100,000

UA (demand only, not clear if necessary outside of j_*): 1-20 [assumes 20 different UA scenarios]

Blood type: 1-3

Center location: 1-4 (based on above model)

DR#1,DR#2: 1-20 × 1-20 [assumes 20 antigen list]

Note: In the actual website these will not be numbers, it will be text boxes or indicators – and will require some additional details for going between something more readable and something that the code can search across.

Create fixed crosswalks:

This can be done in any language because the process only has to be done once.

Transplant center distances

Center 1	Center 2	Distance
1	1	5
1	2	500
1	3	500
1	4	500
2	1	500

{Supplier i, Consumer j}(k) – that is 3 different lists

Supplier ID	Consumer ID	K=0	K=1	K=2
1	1			
1	2			
1	3			
1	4			
2	1			

Create JS script that can find the list $\{i_1, \dots, i_I\}$ based on user input $\{j_*\}$:

This is just a coding task. I let AI do it in the shared program. I use cursor.

Create website that takes user input and outputs details based on above methodology:

This is just a coding task. I let AI do it in the shared program. I use cursor.