Networks and Markets

Hw2 submission

Part 5: Experimental Evaluations

Alon Polski 206530461

Anna Petrenko 320460306

Ariel Chiskis 322442112

**8.**

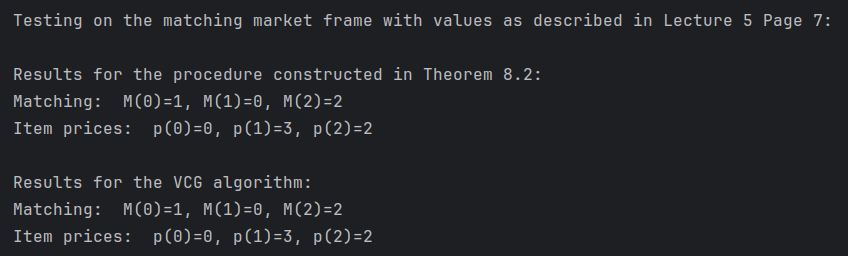
**(b).**

The prices output by the VCG mechanism are identical to the ones output by the

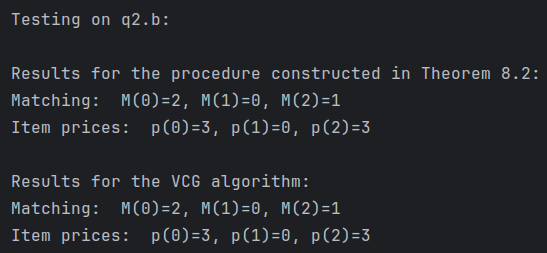
algorithm from Problem 7.

On the matching market frame with values as described in Lecture 5 Page 7

and on the example in q.2 the algorithms output identical prices;

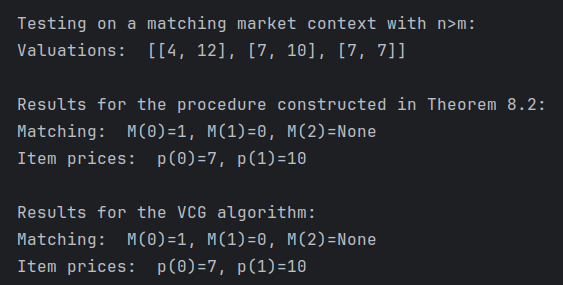


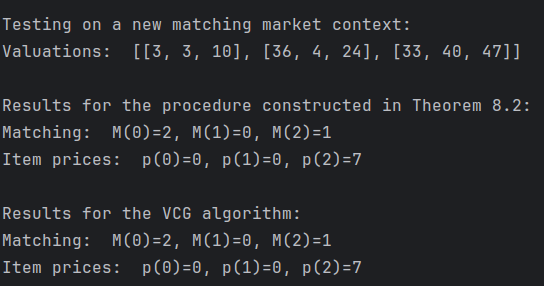
**A black screen with white text

Description automatically generated **

Note, for q2.b the market\_eq() routine chose a market equilibrium matching that differs from the one presented in q.2b solution, both are valid solutions as evident from the final utilites in q.2b.

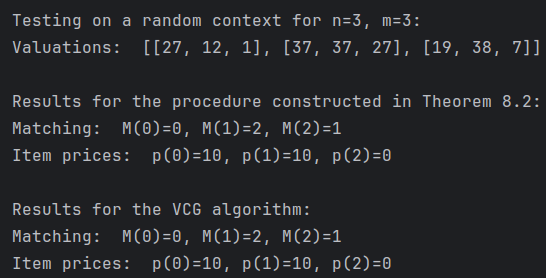
More concrete examples follow.

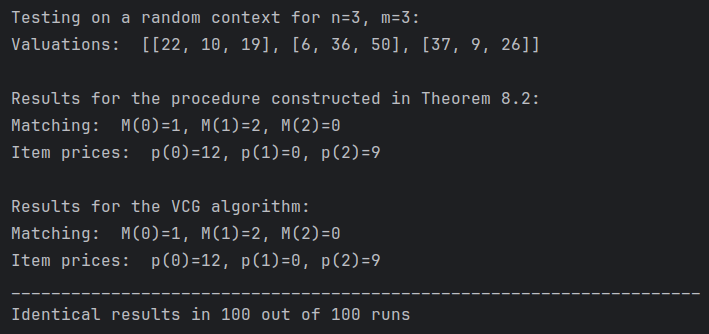


 **A black screen with white text

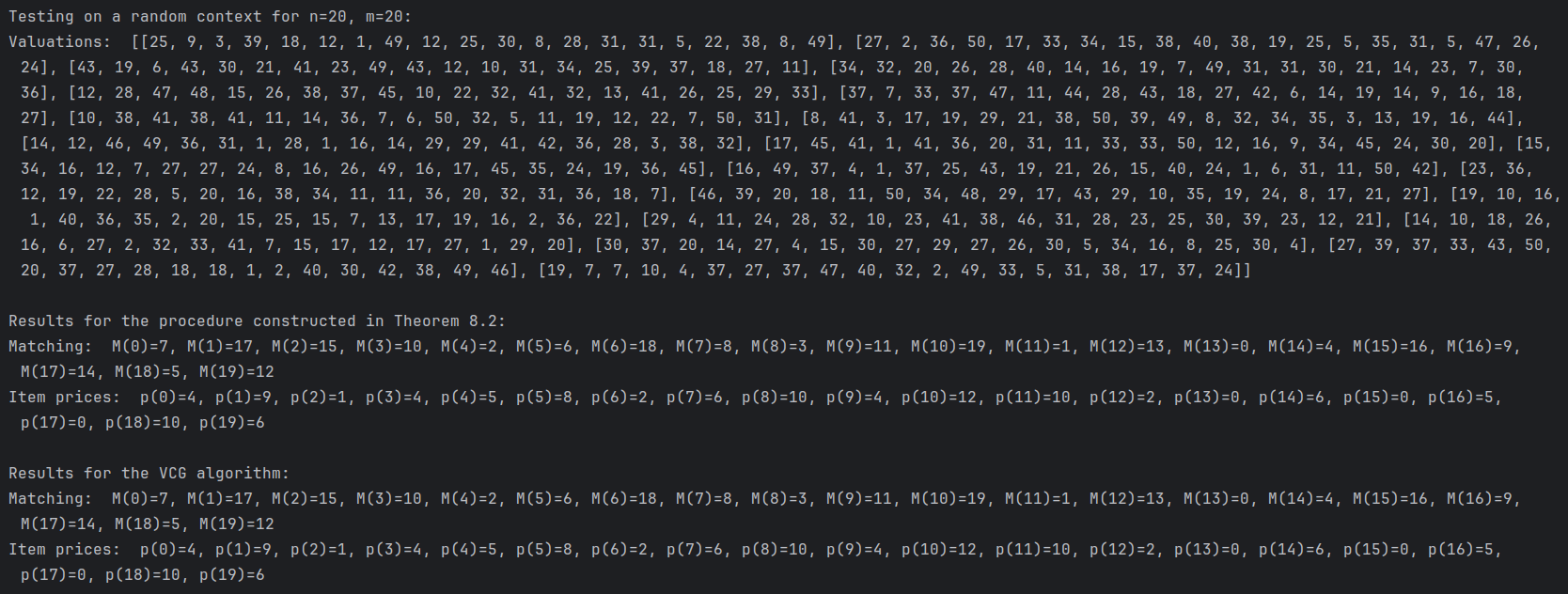
Description automatically generated**

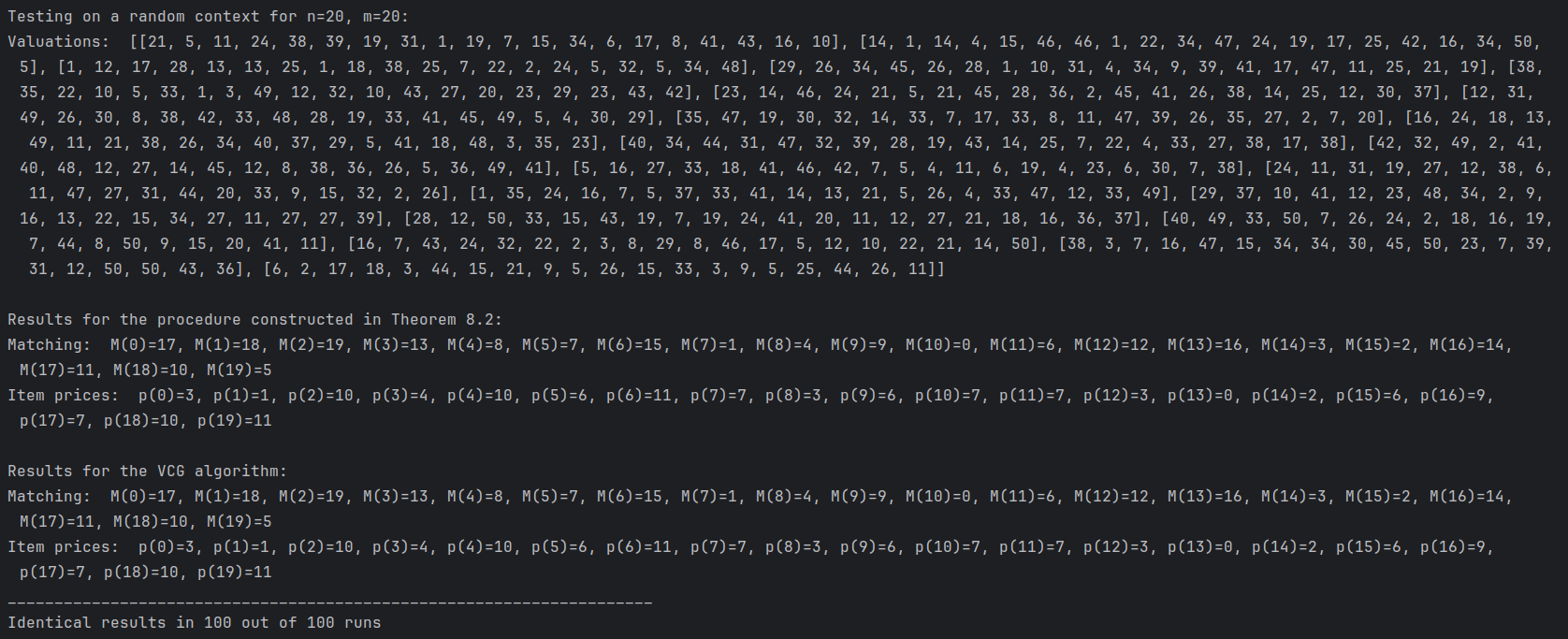
We run both algorithms on random context examples for , the algorithms output identical results in of runs. Observe last two runs and overall comparison;





We run both algorithms on random context examples for , the algorithms output identical results in of runs. Observe last two runs and overall comparison;





**Bonus question 2.**

**(a).** We implemented the bonus routine random\_bundles\_valuations(n,m) and the accompanying run\_vcg\_on\_random\_bundles\_valuations() to run the VCG pricing algorithm on the randomly chosen context.

To simplify the analysis we also implemented the sorted version, random\_bundles\_valuations\_sorted(n, m) similarly generates a matching market context for bundles of identical goods, where each of buyers has a random value for an individual good (between 1 to 50; ties are allowed), but the buyers are sorted in ascending order of value per good s.t the buyer with highest value per good is called buyer no. .

run\_vcg\_on\_random\_bundles\_valuations\_sorted() runs the VCG pricing algorithm on the randomly chosen sorted context, for .

The analysis can be applied w.l.o.g to the sorted version.

As demonstrated in simulation runs below, the results make perfect sense in the bundles context. As expected from the bundles context analysis, the buyer with the highest value per good is matched to the biggest bundle and so on;

A screenshot of a math problem

Description automatically generated with medium confidenceDenoting the buyer ’s valuation for an individual good by and

the number of goods in bundle by ,

in the sorted version of random bundles context,

and as in the analysis in lec.5, an allocation that maximizes SV is

(The largest bundle must go to the person who values the good (and hence the bundle of goods) the most and so on).

This is the matching chosen by the VCG algorithm implementation, complying with the analysis in lec.7.

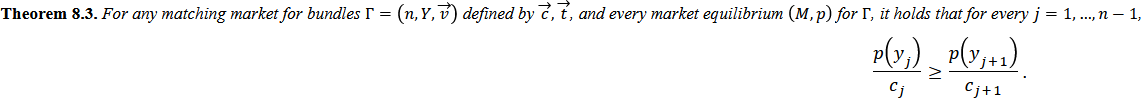
The player with the lowest value per good (player in this context) indeed paid according to our VCG implementation.

A screenshot of a math problem

Description automatically generatedThe prices returned by the VCG alg. are identical to the externality prices computed directly from the analysis in lec.7 as

(note, the affected players are as the buyers are sorted in ascending order).

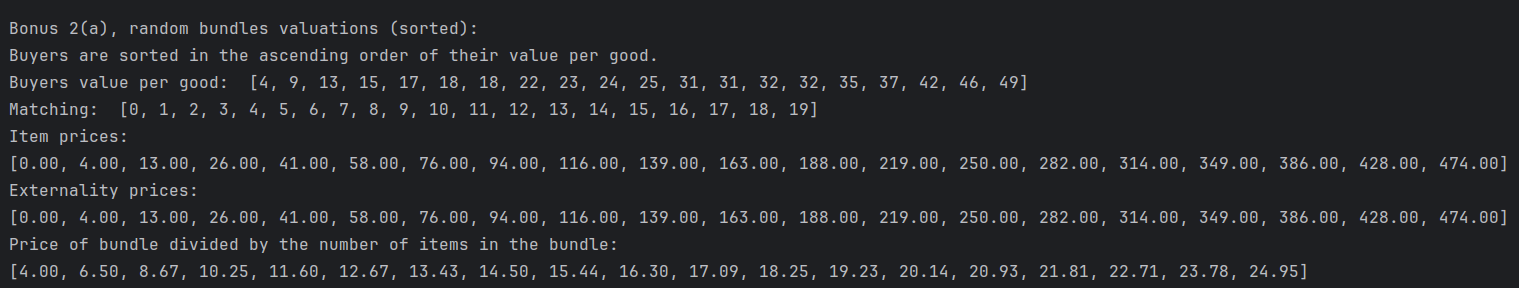
The results also comply with thm8.3;

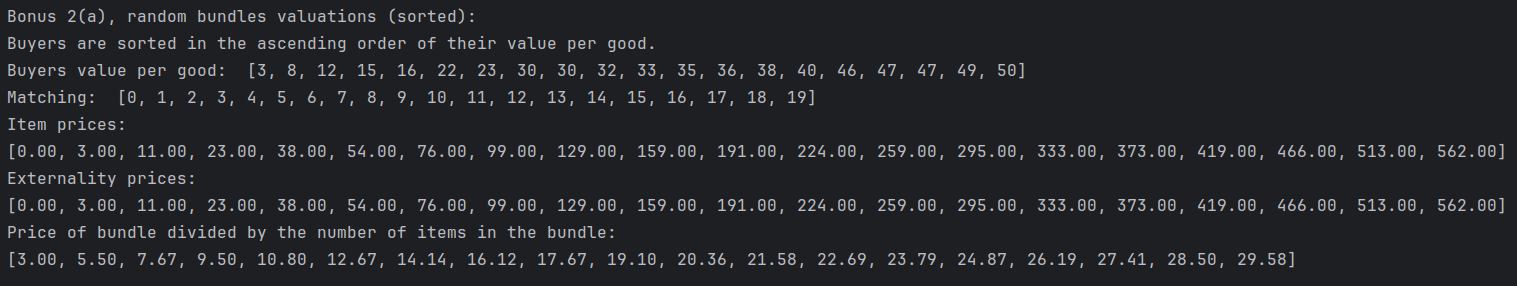


(in reversed order in this context)

This ratio is output in the results for bundle indices .

Several results for analysis follow;





A black background with white and blue text

Description automatically generated

**(b).** When run in the same contexts (with the same randomness), GSP prices are consistently higher than VCG prices. Moreover, the difference increases faster than the bundle size with the GSP price for the bundle of items being almost twice the price of VCG GSP makes people pay the market equilibrium prices.

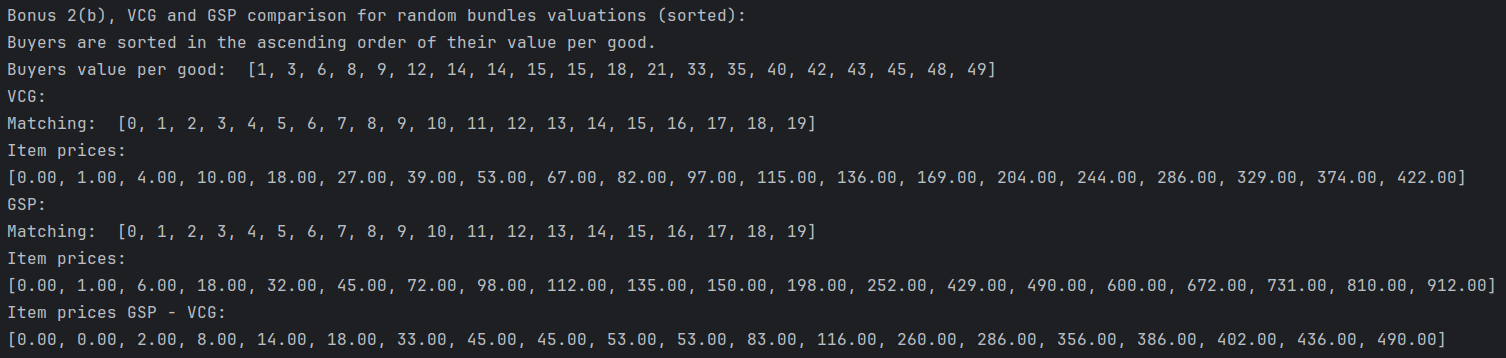
As expected, for both algorithms the lowest price is always .

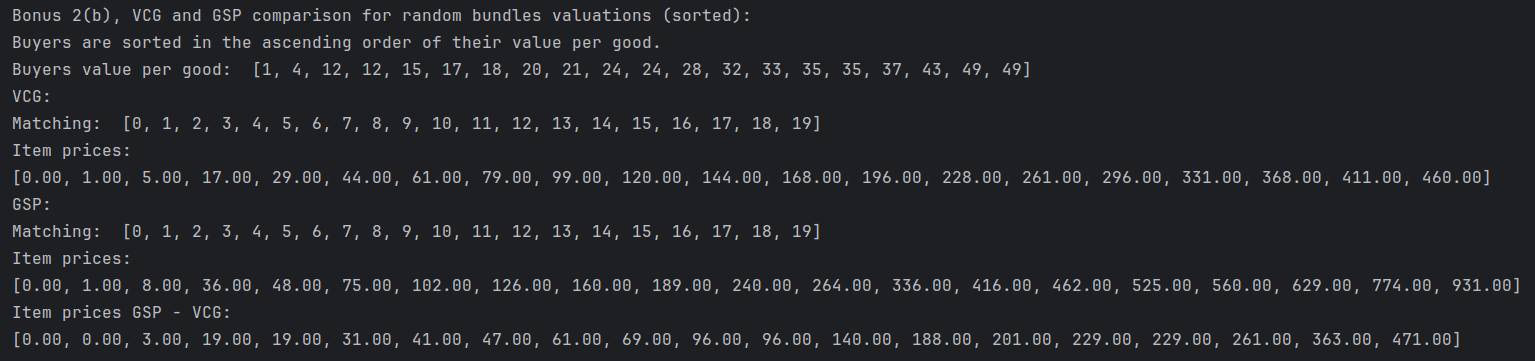
Note that VCG mechanism can DST-implement SV-max for any context,

GSP mechanism is not DST, but it Nash-implements SV-max.

תמונה שמכילה טקסט, צילום מסך, גופן

התיאור נוצר באופן אוטומטי





A computer screen shot of numbers

Description automatically generated

A screen shot of a computer screen

Description automatically generated

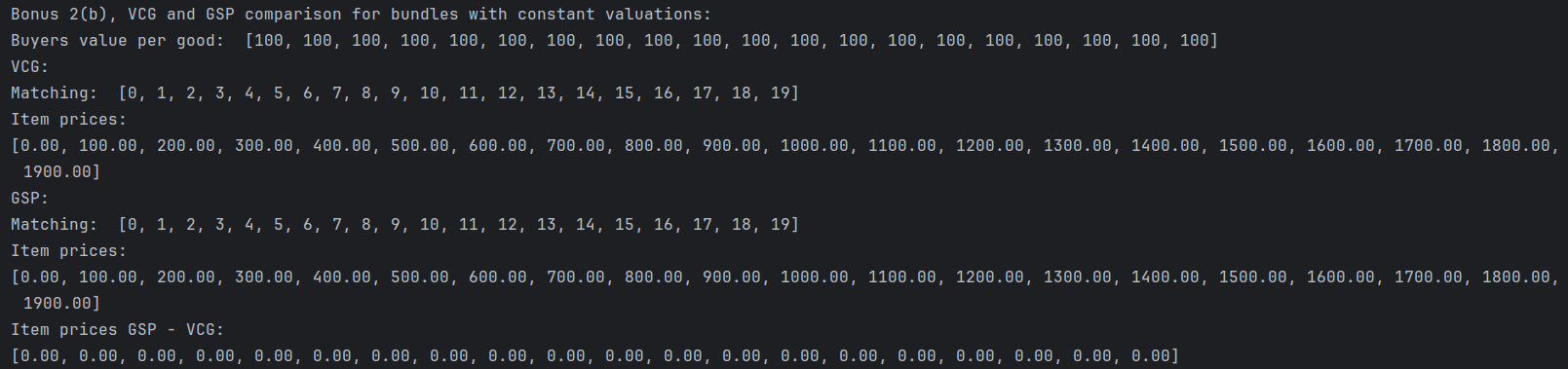
תמונה שמכילה טקסט, צילום מסך, גופן

התיאור נוצר באופן אוטומטי

Interestingly, for a context where all players have identical value per good, both algorithms return identical results.

תמונה שמכילה צילום מסך, טקסט, גופן

התיאור נוצר באופן אוטומטי



**9.** We think the most logical thing would be to define the value on the edge between rider and driver as:  
if that value is positive, otherwise the value will be .  
That's because it seems that in an exchange network setting, it's reasonable to view the value on the edge between rider and driver as the value driver has for rider 's ride and can capture that value because the driver will get paid for giving that ride, however, the driver will have to travel from his location to the rider's location and then from that location to the rider's destination, which decreases his revenue by a function of the distance travelled (because it wastes fuel and time.), this also decreases rider 's value for the match due to time wastage.  
The reason we added that zero evaluation when is negative is [this discussion on the forums.](https://moodle.tau.ac.il/mod/forum/discuss.php?d=96767)

**10b.** The results are as follows:

* For 10 riders and 10 riders:  
  Average rider profits: 16.485  
  Average price: 2.131
* For 20 riders and 5 drivers:  
  Average rider profits: 4.629  
  Average price: 19.706
* For 5 riders and 20 drivers:  
  Average rider profits: 26.284  
  Average price: 0.312

(The way we calculated rider's profits is their utility in the matching markets reduction, the way we calculated price is utility of a matched drivers. Because it would be weird to consider the utility of an unmatched driver a price.)

First, we can notice that if there are much more drivers than riders, riders will get more profits than if there were the same number of riders and drivers, also in that case riders will get more profits than the drivers.   
When there are more riders than drivers, riders will get even less value than if there were the same number of riders and drivers and riders will earn less than drivers. The same is true vice versa;  
If there are more riders than drivers, drivers will get more value than if there is the same number of riders and drivers. When there are more drivers than riders, drivers will get even less value than if there was the same number of drivers and riders.  
That seems like a very truthful representation of a property of real markets, as supply grows bigger than the demand, prices will decrease, and the consumers will have the upper hand. As supply becomes smaller than demand, prices grow and sellers will have the upper hand.  
Second interesting phenomenon we can notice is that when there is the same number of drivers and riders, riders will earn more than drivers, also, riders earned more when there were 5 riders than how much drivers earned when there were 5 drivers, and also, riders earned more when there were 20 riders than how much drivers earned when there were 20 drivers, this backs up the suggestion from the answer to question 2(c), the says that probably the market equilibrium algorithm prioritizes the buyers.

**11.** We can multiply a rider's value, , by a factor, , and replace a rider's value by .  
 should be a function of the rider's location and destination, perhaps a sum , where is determined by the rider's location and is determined by the rider's destination. will reflect how popular are the rider's location and destination, for example, if the driver is heading to an airport will be big, and if he's heading to a derelict suburb, will be small, will behave similarly.  
That way, when a rider is heading to or located at a location that is popular among drivers, the value on his edges in the exchange network would be higher, that way, he'll be more likely to get a match and a higher price.