

WEB – BASED OFFICE MARKET

SJSU SAN JOSÉ STATE
UNIVERSITY

Advisor

Dr. Chris Pollett

Committee Members

Dr. Robert Chun

Mr. Rajesh Krishnan

By

Manodivya Kathiravan



Agenda

- Introduction
- Bartering System
- Problem Statement
- Preliminary Work Summarized
- Multi-Bartering exchange System
- Experiments – Testing
 - Areas of Improvement
 - Conclusion
 - Demo

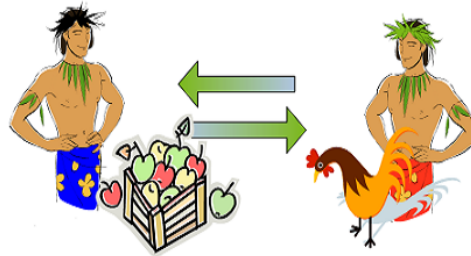
About the Project

- People who work in an office often have different pools of resources that they want to exchange
- They want to trade their resources(seller) with a person who wants that particular resource(buyer) and in return get another resource the buyer offers.
- Barter-exchange : Where an item is traded for another item without the involvement of actual money

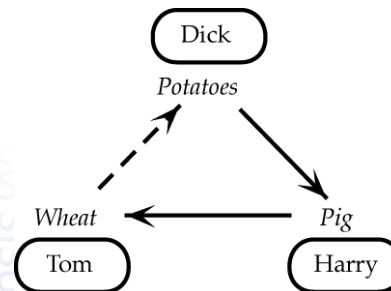


About the Project(Contd..)

- Direct - An exchange is complete when there is a match between an available item and a desired item



- Multi-lateral - Item desired by one user is made available through a series of intermediate exchanges



- I will be working to design an online bartering system with multi-lateral possibilities.

Introduction

- Old method of exchange since eighteenth century



- Olden days bartering was done between users around the same locality. Now its done globally
- Main Advantage : No use of money
- Mostly this kind of trading are done through swap markets and online auctioning.

Introduction(Contd..)

- Set of mathematical problems called matching problem that has been on focus in mathematics and research for some time.
- In any matching problem there is a group of objects.
- The idea is to connect the objects in a pair-wise fashion with one another.
- Consider couples that come to a dance. Imagine that each of the couples is married, and a person distributes cards that specify who will dance with whom.
- If two of these dancing assignments result in a couple who is also married, we call this a match.

Problem Statement

- The search in mathematics is to find an algorithm that will efficiently compute the maximum number of pairings or matches that can be produced from given a set of objects
- We need to find the maximum pairings from the given set of exchange pool.
- Vertices in a graph represent the set of objects. The pairings are represented by edges connecting two objects.
- No edge can coincide with more than two vertices, one at each end of the edge.
- Using brute-force approach to find out the match it will not be practical.
- The leading question for this project is how efficiently a possible multi – barter scenario can be determined from given set of objects.

Preliminary Work Summary

- Sonmez et al [1] mentions Multi-way trading falls under the rubric of a domain of mathematics called matching theory.
- A typical problem that arises in matching theory is the housing allocation problem.
- Set of agents are to be assigned to a set of houses
- Each agent has a preference for the first house they desire, the second, and so on.
- The goal is to assign the house that is highest on the agent's preference list from the available houses.
- As time goes, houses that are at the top of an agent's preference list may become unavailable and therefore results in an assignment further down on the agent's preference list.

Preliminary Work Summary (Contd.)

- One variation of this house assignment problem is to add to the problem that each agent wants a house but also has a house to trade.
- Out of matching theory comes the algorithm to fulfill the desired trades.
- Serial dictatorship algorithm - an agent is assigned a priority according to some protocol defined by a function.
- The agent with the highest priority as determined by the function will be assigned their top choice house using their list of preferences.
- Subsequent agent will be selected and assigned a house in a similar fashion
- Housing problem applicable to bartering – the assignment of desired resources to agents that desire them.

Preliminary Work Summary (Contd.)

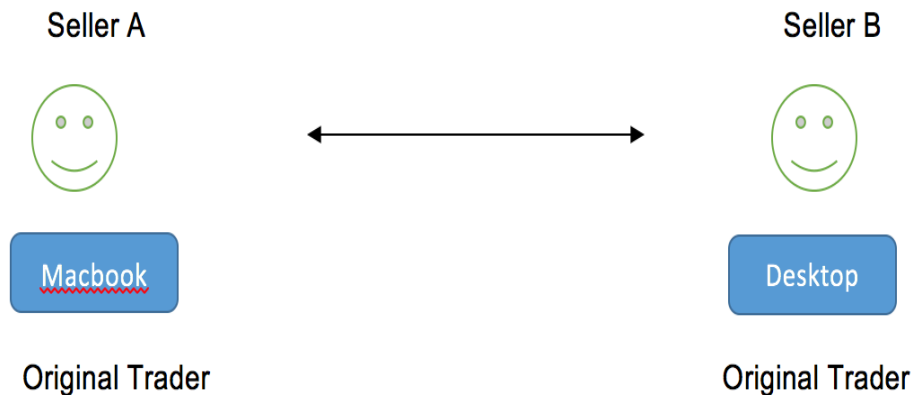
- The concept of trading sequences in multi- way trading comes to play in matching theory when we consider another classic example, the stable marriage problem by Halldorsson et al. [2]
- Identical number of men and women. Each has a preference list of whom they wish to marry of the opposite sex.
- Important to identify a maximal matching such that the highest number of pairs of men and women are matched
- Stable if there are no blocking pairs in it
- Can be ties - when two or more agents make the same selection.
- Rearranging the ties in such a way as to create new lists of preferences.
- If we consider that a trade preference is like a partner preference, then we can see how the algorithm for computing a stable marriage might be used to compute possible trades.

Preliminary Work Summary (Contd.)

- Another practical application is in kidney exchange by Abraham et al. [5]
- There is a pool of donors and recipients
- if a one-to- one match can be made, then the recipient will receive the donor's kidney
- There are times when an assignment cannot be made - multi-way matching
- Current popular online sites - Craigslist, uExchange
- The algorithm they have find direct matches quickly and efficiently whereas they don't have any kind of multi-barter possibilities
- Too many users in the pool offer the same exchange then there has to be a way to pick one

MULTI-BARTERING SYSTEM

- Seller A is willing to offer a MacBook and desires a desktop
- Seller B is willing to offer a desktop for a MacBook
- If no such sequence is found - all exchange sequences of length 3 will be considered
- exchange sequence is found or until a threshold is reached



ALGORITHM:

$$S = \{ \langle O_{d1}, O_{o2} \rangle, \langle O_{d2}, O_{o3} \rangle, \langle O_{d3}, O_{o4} \rangle \dots \langle O_{dn-1}, O_{o1} \rangle \}$$

- O_d -> desired/required item by the trader
- O_o -> object willing to trade
- Desired trade = $\langle O_d, O_o \rangle$

Steps:

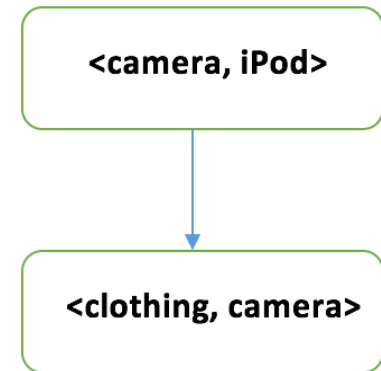
1. Set O_{df} to O_d
2. Create a tree rooted at O_{df}
3. find all p such that $O_{df} = O_{op}$ such that $\langle O_{dp}, O_{op} \rangle$
4. If p is found ->
 - Attach p to current node
 - Check if $O_o = O_{dp}$, if yes sequence is found and terminate
 - otherwise Set $O_{df} = O_{dp}$ and go to step 3
5. If p not found ->
 - No sequence is found

Multi- Bartering system (Contd.)

$P = \{ \langle \text{camera}, \text{iPod} \rangle, \langle \text{iPod}, \text{shoes} \rangle, \langle \text{shoes}, \text{clothing} \rangle, \langle \text{clothing}, \text{camera} \rangle, \langle \text{shoes}, \text{flashlight} \rangle, \langle \text{flashlight}, \text{radio} \rangle \}$.

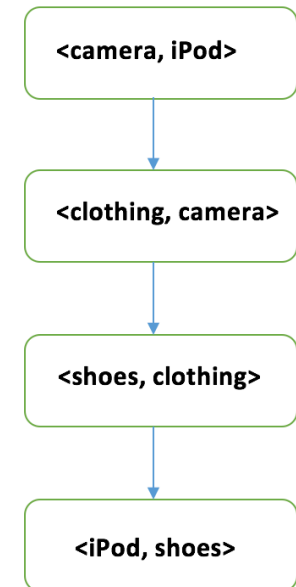
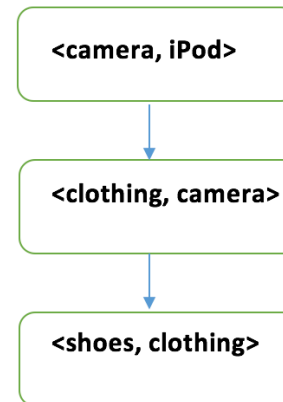
Desired exchange : $\langle O_d, O_o \rangle = \langle \text{camera}, \text{iPod} \rangle$

- Set $O_{df} = \text{camera}$
- We begin by creating a tree with root node O_{df}
- Then, we find all p such that $O_{df} = O_{op}$
- Note that we are looking for all p such that the offered object of the pair is the same as the desired object of the current node (which initially is the root node)
- Only one relevant $\langle \text{clothing}, \text{camera} \rangle$.



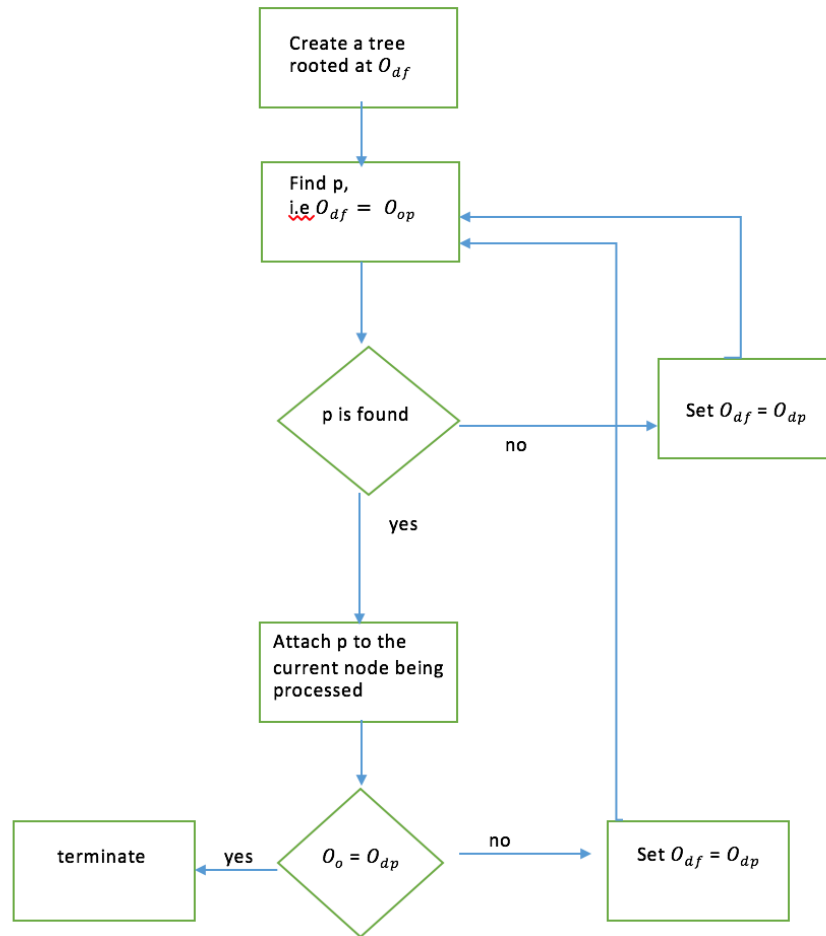
Multi- Bartering system (Contd.)

- Now we Set $O_{df} = \text{Clothing}$
- Then, we find all p such that $O_{df} = O_{op}$
- We set $O_{df} = \text{Shoes}$
- At this point $O_o = O_{dp}$, where $O_o = \text{iPod}$ and also $O_{dp} = \text{iPod}$, then a path has been found that offers the originally desired object.



Multi- Bartering system (Contd.)

Activity Diagram showing the flow of algorithm



Multi- Bartering system (Contd.)

Algorithm Analysis

- In a brute-force approach to multi-way trade, no heuristics are used to reduce the search space.
- The approach is to blindly create a tree of all trades, regardless of whether the trades are valid or not.
- The root of the tree is the desired trade. To this root we attach all other trading pairs that are in the trading pool.
- Consider n trading pairs in the trading pool
- First level of the tree $\rightarrow n$ possible trades to extend.
- Likewise, if each node at each level is extended in this way, to each of the first level's
- Each child pair in the first level will have n children pairs in the second level.
- Since the first level will have n children, in the second level their will be n children times n pairs, or n^2 nodes

Multi- Bartering system (Contd.)

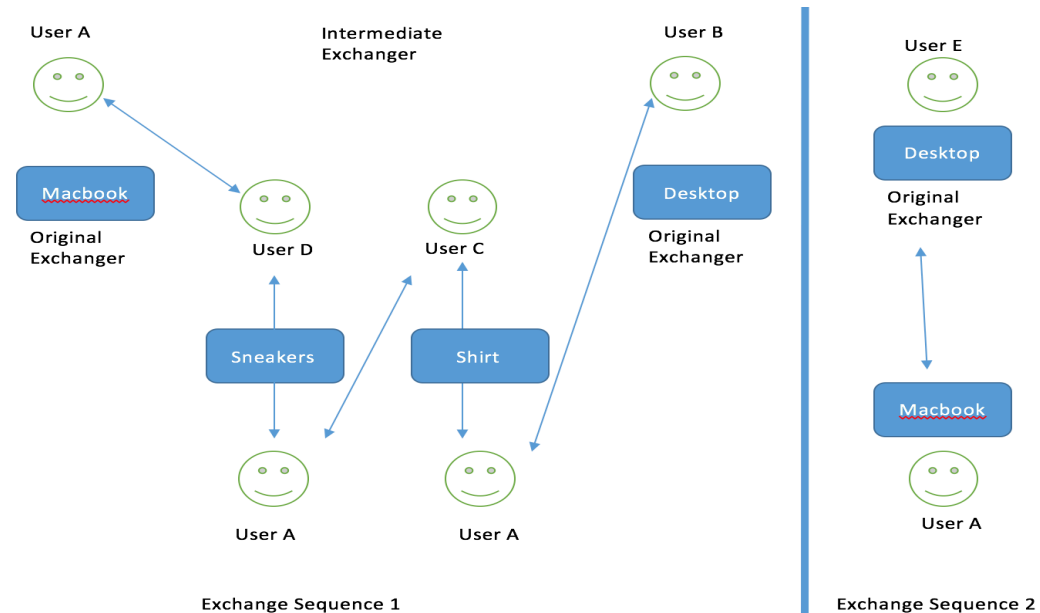
Algorithm Analysis

- Thus now there will be n nodes and n^2 and the total number of nodes in the tree will be $1 + n + n^2$ nodes
- The total number of nodes depend on height of the tree h needed to find the trading sequence that will satisfy the original trading pair
- Performance of the brute-force algorithm is calculated to be $O(n^n)$. The number of nodes in the tree that will have to be examined will be of the order n^n .

Three Heuristics

Minimum Length Exchange Sequence

- Desired exchange with the most minimum possible length
- If there are more than one way to facilitate the necessary exchange the algorithm must be efficient enough to pick the shortest path in the exchange instead of the longest path.

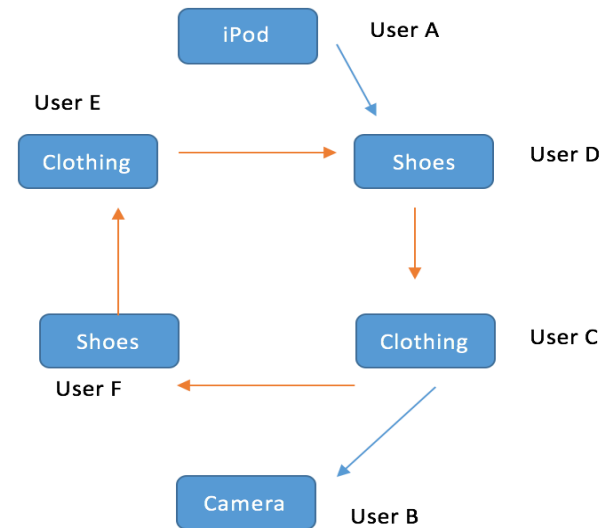


Three Heuristics (Contd.)

Avoid Cycles in the Exchange Sequence

- Cycles must be avoided when looking for the exchange sequence.
- When we use brute-force methodology to find the desired multi way exchange there is a possibility for the exchanges to go into a cycle

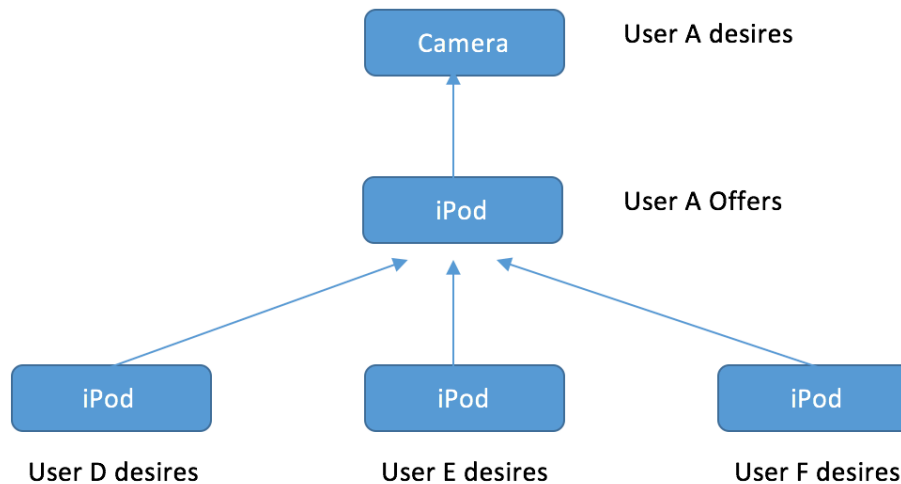
User	Desires	Willing to Offer
A	Camera	iPod
B	Clothes	Camera
C	Shoes	Clothes
D	iPod	Shoes
E	Shoes	Clothes
F	Clothes	Shoes



Three Heuristics (Contd.)

Satisfy the needs of Current Exchange Pair

- If there are more than one user desiring for the same item, then the algorithm will consider the user that closely satisfy the needs for the current exchange pair
- The algorithm will pick the desired user based on what they have to trade and whether that item can complete the exchange sequence.



Multi- Bartering system (Contd.)

Complexity for Multi-barter system

- single root node consisting of the desired trade pair
- In the worst case there will be $(n - 1)$ trading pairs that may be relevant if all of the remaining trading pairs in the trading pool are relevant
- This means that the root of the tree will have $(n - 1)$ children
- Once again we assume that, for each of the trading pairs that constitute the first level of the tree, there are $(n - 2)$ possible trading pairs that could serve as the children of each of these nodes
- Tree is being built in a breadth-first fashion, when a trading pair satisfies the original trading pair, the algorithm can terminate.

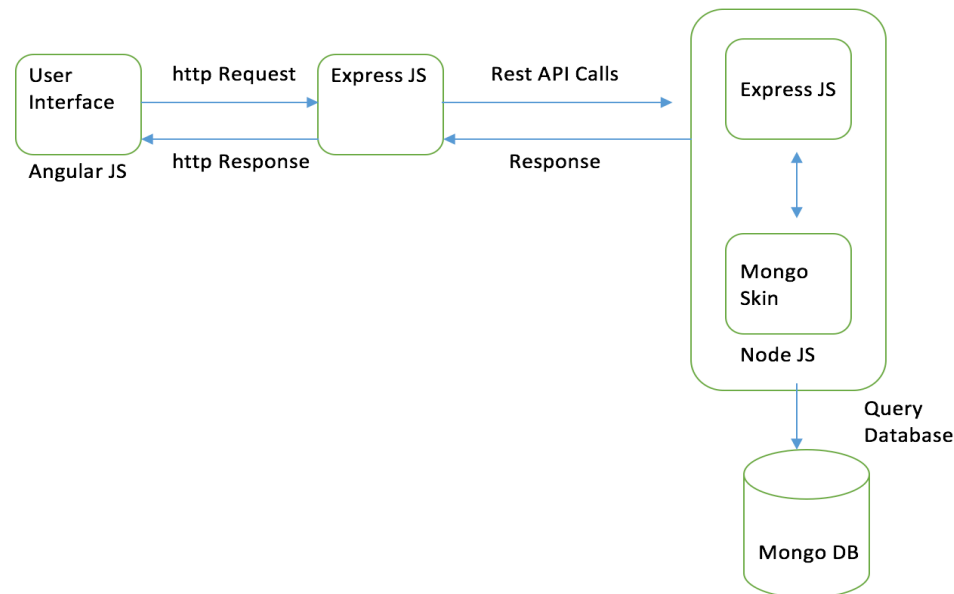
Multi- Bartering system (Contd.)

Complexity for Multi-barter system

- Again, if we consider that all of the trading pairs will be exhausted in the search, the length of the trading sequence will be n , the number of trading pairs in the tree. Then that we have the following complexity of the tree, counting the trading pairs in the tree:
$$1 + (n - 1) + [(n - 1) (n - 2)] + [(n - 1) (n - 2) (n - 3)] + \dots$$
- In the case of the multi-way trading algorithm presented here, the performance is much better for the following reason:
 - the only nodes that are used are those that are relevant to a particular step in the trading sequence

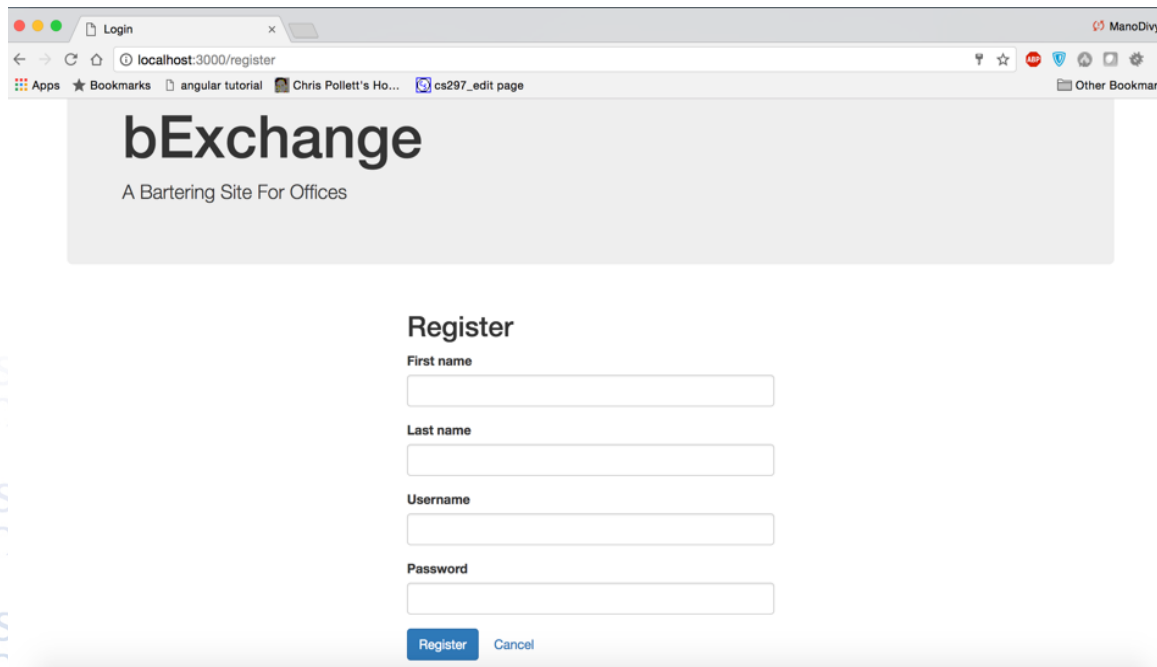
Implementation Stack

- Angular JS is used on the client side/ front-end of the MEAN stack.
- Node JS is used on the server side.
- Express JS is used as a node JS web application framework.
- Mongo DB is a no SQL database that is scalable and uses JSON like documents



Registration Module

- split the login and registration pages out from the angular application
- secure access to the angular client files, so all front end angular files (including JavaScript, CSS, images etc.) are only available to authenticated users.



The screenshot shows a web browser window with the address bar displaying 'localhost:3000/register'. The page features a header with the 'bExchange' logo and the tagline 'A Bartering Site For Offices'. Below the header, there is a 'Register' section with four input fields: 'First name', 'Last name', 'Username', and 'Password'. At the bottom of the registration form, there are two buttons: 'Register' and 'Cancel'.

Add Item Module

- The add item module will add an item to the exchange pool
- The exchange information is stored in a JSON object and a POST API request is made to the server

bExchange x Mano

localhost:3000/app/#/addItem

Apps Bookmarks angular tutorial Chris Pollett's Ho... cs297_edit page Other Book

Home Account **Add an Item** Show Items View Trades Logout

Item Added successfully

Add An Item

What I offer

ipod

What I look for

shoes

Save Cancel

Show Items Module

- The show items module displays all the available exchanges from the exchange pool.
- A GET request is made to the server to fetch all the necessary information.

Item No	Seeking	Offering	Actions
1	book	car	<button>Barter</button>
2	ipod	shoes	<button>Barter</button>
3	b	a	<button>Barter</button>
4	hat	bat	<button>Barter</button>
5	strawberry	rasberry	<button>Barter</button>
6	shoes	ipod	<button>Barter</button>

Username	Seeking	Offering
orange	asasas	asd
orange	asasasasassa	aassaa
orange	asasasasassa	aassaa
orange	camera	mobile
orange	camera	mobile
orange	camera	mobile
orange	car	book
orange	note	glass
orange	shoes	clothing
orange	macbook	samsungphone
orange	a	c
orange	cat	dog

Barter Module

- The barter module posts the item the user is looking to barter as an JSON object to the server

The screenshot shows a web browser window titled "bExchange" with the URL "localhost:3000/app/#/showItems". The browser displays a table of items for sale:

DaveMini	macbook	samsungphone
DaveMini	samsungphone	macbook
mikeRed	Lens	Camera
mikeRed	bat	cat
yoda	blackberry	strawberry
StuartMinion	clothing	camera
StuartMinion	rasberry	blackberry
StuartMinion	pen	paper

Below the table, the section "Matches Found" is visible. It includes a "View Steps" button and a list of steps:

1. Kevin can exchange book for a car with Dave

A table shows the details of the match:

Seeking	Offering	User Rating
car	book	1

At the bottom, a message states "An exchange is found!!" and asks "Would you like to contact the user and proceed with the exchange?" with a green "Yes" button.

View Trades Module

- listing the pending trades of the user
- User Rating =
$$\frac{\text{Total number of Transactions}}{\text{Total Number of successful transactions}}$$

The screenshot shows a web browser window with the address bar displaying 'localhost:3000/app/#/viewTrades'. The page has a navigation bar with links: Home, Account, Add an Item, Show Items, View Trades (highlighted), and Logout. Below the navigation bar, there are two sections:

My Current traded Items

Item No	Seeking	Offering	Actions
1	b	a	<button>Complete</button>

My Completed Trade Items

Item No	Seeking	Offering
1	book	car

Experiments –Testing

These experiments are divided into three parts:

- Experiments conducted to unit test all the features in the barter system
- Experiments to stress test the system and find out how it reacts to different load of data
- A/B testing the system

Experiments – A/B Testing

- This testing, also known as split testing, involves comparing two different versions of a web page to see which one performs better.
- The two variants are shown to similar visitors and their responses are recorded. The one that provides a better conversion rate, wins.

Matches Found

View Steps

1. Kevin can exchange raspberry for a blackberry with Dave
2. Kevin can exchange blackberry for a strawberry with Stuart

Seeking	Offering	User Rating
blackberry	strawberry	1
rasberry	blackberry	1

An exchange is found!!

Would you like to contact the user and proceed with the exchange ?

Yes

Variant A

Matches Found

Seeking	Offering	User Rating
blackberry	strawberry	1
rasberry	blackberry	1

An exchange is found!!

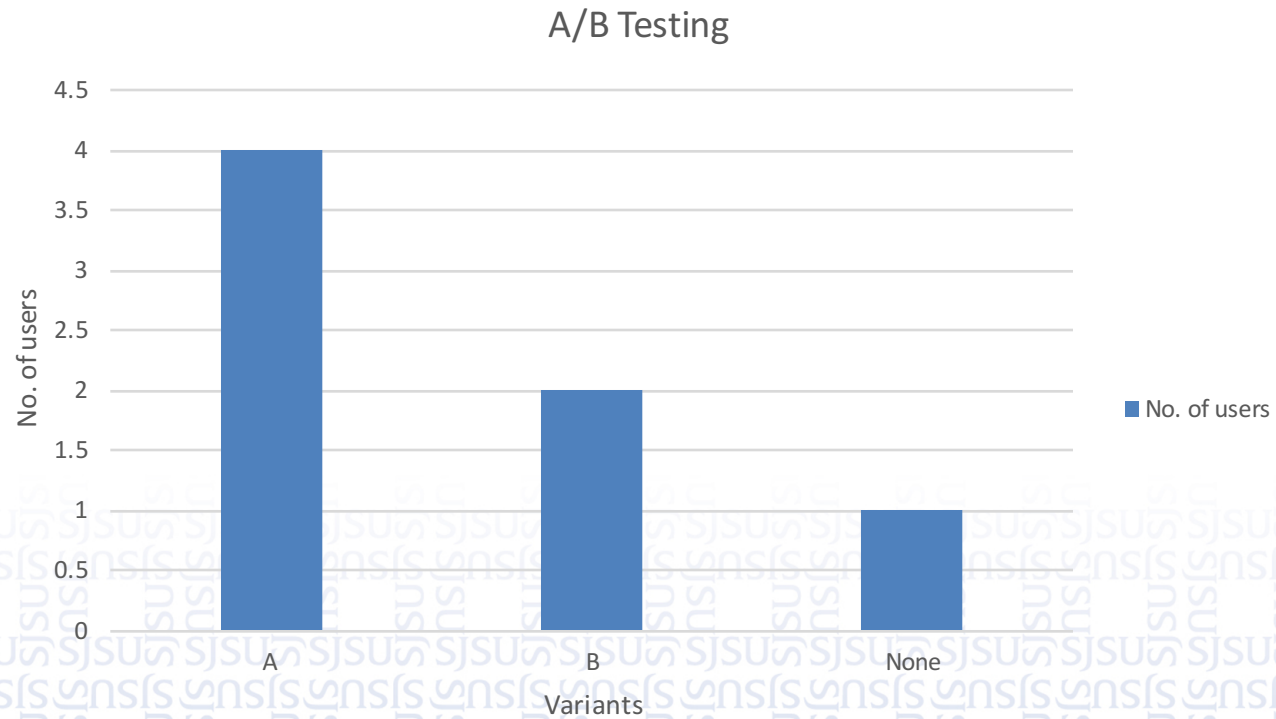
Would you like to contact the user and proceed with the exchange ?

Yes

Variant B

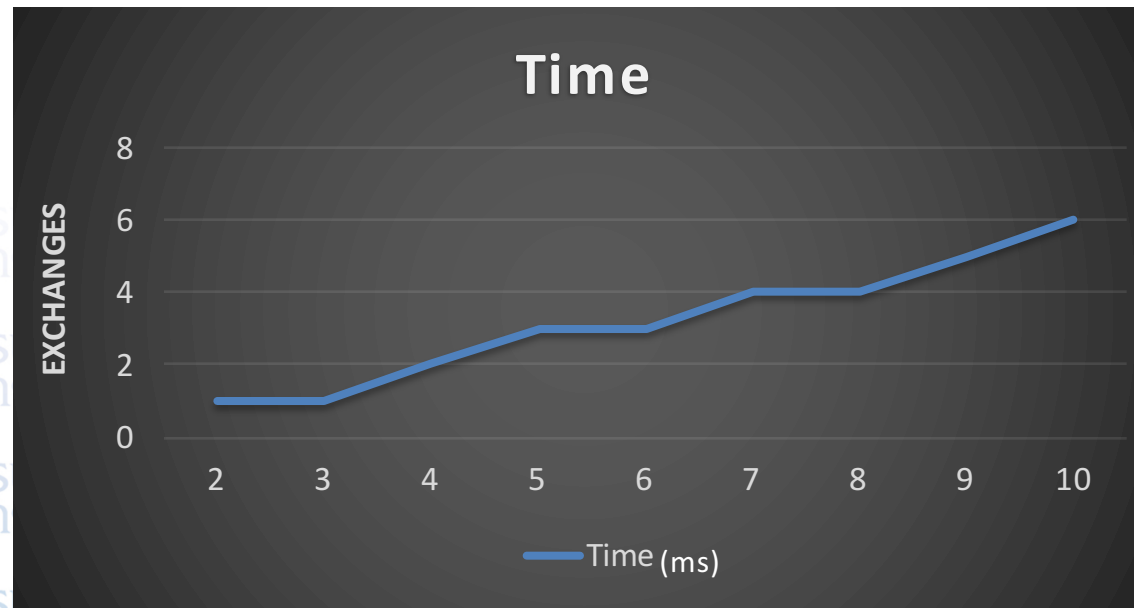
Experiments – A/B Testing (Contd.)

The results of A/B testing are as follows:



Experiments – Stress Testing

- Stress testing also called torture testing is to test the system beyond operational capability and look for results.
- In this part of the experiment we start with 500 exchanges made available in the exchange pool and start with direct barter scenario i.e. exchanges = 2 and we go about increasing the node size thereby making the system to look for multi barter scenarios and test the system for stability and correctness of the result.



Example 1

Matches Found

View Steps

1. Kevin can exchange raspberry for a blackberry with Dave
2. Kevin can exchange blackberry for a strawberry with Stuart

Seeking	Offering	User Rating
blackberry	strawberry	1
rasberry	blackberry	1

An exchange is found!!

Would you like to contact the user and proceed with the exchange ?

Yes

Example 2

Matches Found

View Steps

1. A can exchange Ipod for Sneakers with E
2. A can exchange Sneakers for Shoes with D
3. A can exchange Shoes for clothes with C
4. A can exchange clothes for a Camera with B

Seeking	Offering	User Rating
Ipod	Sneakers	1
Sneakers	Shoes	1
Shoes	Clothes	1
Clothing	Camera	1

An exchange is found!!

Would you like to contact the user and proceed with the exchange ?

Yes

Example 3

Matches Found

View Steps

1. A can exchange Ipod for book with K
2. A can exchange book for pant with I
3. A can exchange pant for shirt with H
4. A can exchange tshirt for shirt with G
5. A can exchange tshirt for socks with F
6. A can exchange socks for Sneakers with E
7. A can exchange Sneakers for Shoes with D
8. A can exchange Shoes for clothes with C
9. A can exchange clothes for a Camera with B

Seeking	Offering	User Rating
Ipod	book	1
book	pant	1
pant	shirt	1
shirt	tshirt	1
tshirt	socks	1
socks	sneakers	1
Sneakers	Shoes	1
Shoes	Clothes	1
Clothing	Camera	1

An exchange is found!!

Would you like to contact the user and proceed with the exchange ?

Yes

Experiments – Unit Testing

- The experiments conducted to unit test the features in the barter system gives us an idea on how well the system behaves under given conditions
- **Testing a Controller :**
 - *angular.mock.inject*
 - *beforeEach*

```
describe('PasswordController', function() {
  beforeEach(module('app'));

  var $controller;

  beforeEach(inject(function(_$controller_){
    // The injector unwraps the underscores (_) from around the parameter names when matching
    $controller = _$controller_;
  }));

  describe('$scope.grade', function() {
    it('sets the strength to "strong" if the password length is >8 chars', function() {
      var $scope = {};
      var controller = $controller('PasswordController', { $scope: $scope });
      $scope.password = 'longerthaneightchars';
      $scope.grade();
      expect($scope.strength).toEqual('strong');
    });
  });
});
```

Experiments – Unit Testing

- The experiments conducted to unit test the features in the barter system gives us an idea on how well the system behaves under given conditions
- **Testing a Controller :**
 - *angular.mock.inject*
 - *beforeEach*

```
describe('PasswordController', function() {
  beforeEach(module('app'));

  var $controller;

  beforeEach(inject(function(_$controller_){
    // The injector unwraps the underscores (_) from around the parameter names when matching
    $controller = _$controller_;
  }));

  describe('$scope.grade', function() {
    it('sets the strength to "strong" if the password length is >8 chars', function() {
      var $scope = {};
      var controller = $controller('PasswordController', { $scope: $scope });
      $scope.password = 'longerthaneightchars';
      $scope.grade();
      expect($scope.strength).toEqual('strong');
    });
  });
});
```

Experiments – Unit Testing

- **Testing an API Call**

- When performing unit tests we want our tests to quickly run without having any external dependencies such as JSONP or XHR requests to actual server
- \$http service
- \$http service uses \$httpBackend service to send these requests
- \$httpBackend.expect – This indicates what a http request expects
- \$httpBackend.when – This indicates what definition is needed by the backend

```
it('should fail authentication', function() {  
    authRequestHandler.respond(401, '');  
  
    $httpBackend.expectGET('/login');  
    var controller = createController();  
    $httpBackend.flush();  
    expect($rootScope.status).toBe('Failed...');  
});
```

Areas of Improvement

- Currently, the system works for multi barter scenarios without any quantity specified.
- A future improvement to the project can be to add a module that can handle quantities of items.
- Given a quantity of item the system must be able to find a direct matching exchange or substantially reduce the quantity of the item and look for bartering possibilities.
- This reduction value can either be determined by the user or we can use systems such as auctioning systems and set a time and minimum value the user is ready to offer

Conclusion

- In this project, we designed an online barter exchange system similar to existing systems such as craigslist, uExchange called bExchange which has the feature to perform multi-lateral or indirect bartering exchanges
- The main advantage of this system is it starts with the search for direct exchange for any given exchange scenario, incase if its not possible for a direct exchange it looks for a multi-bartering possibilities from the exchange pool.
- The system also provides every user to have a user rating based on the success with older exchanges
- This rating will be very useful when there are more than one users offering the same exchange.
- The system works efficiently to find the minimum length sequence for any given exchange. It also avoids any kinds of cycles in exchange sequence.

References

- [1] Barter system (2015) Retrieved from <https://en.wikipedia.org/wiki/Barter>
- [2] Edmonds, J. Paths, trees, and flowers.1965. Canadian Journal of Mathematics, 449–467.
- [3] Sonmez, T., and Unver, M. U. 2009. Matching, allocation, and exchange of discrete resources., Boston, MA.
- [4] Halldorsson, M. M., Iwama, K., Miyazaki, S., and Yanagishi. 2007. H. Improved approximation results for the stable marriage problem. In proceedings on the ACM Transactions on Algorithms.
- [5] Abraham, D. J., Blum, A., and Sandholm, T. Clearing algorithms for barter exchange markets: Enabling nationwide kidney exchanges. In Proceedings of the 8th ACM Conference on Electronic Commerce, San Diego, CA, ACM Press, New York, NY, 2007.
- [6] Unver.2010. M. U. Dynamic kidney exchange. Review of Economic Studies, 372–414.

References (Contd.)

[7] Vries, S. D., and Vohra, R. 2003. Combinatorial auctions: a survey. *INFORMS Journal on Computing*, 284–309.

[8] Vijay Krishna .2010. *Auction Theory*. Academic Press.

[9] Randy.M. Kaplan. 2011. An improved algorithm for multi-way trading for exchange and barter. *Electronic Commerce Research and Applications* Volume 10, Issue 1, Pages 67–74

[10] How Barter system works retrieved from
<https://www.mint.com/barter-system-history-the-past-and-present>

[11] Sebastien Mathieu. 2006. Match-Making in Bartering Scenarios. Master's thesis. University of new brunswick, Canada

Demo

SAN JOSÉ STATE UNIVERSITY *powering* SILICON VALLEY



Questions ??

SAN JOSÉ STATE UNIVERSITY *powering* SILICON VALLEY



Thank You!