Yale SCHOOL *of* MANAGEMENT

YALE UNIVERSITY
School of Public Health

YALE UNIVERSITY
School of Engineering and
Applied Science

Physical Flow Approximations for the FCFS Stochastic Matching Model

Mohammad Fazel-Zarandi
Post-Doctoral Researcher, Yale School of Management

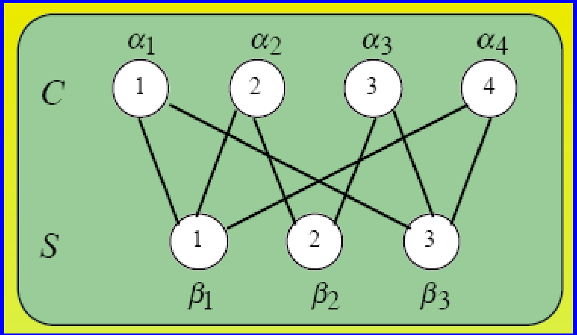
Edward H. Kaplan
William N and Marie A Beach Professor of Operations Research, Professor of Public Health, Professor of Engineering

Yale University
New Haven, Connecticut

2017 LNMB Lunteren Conference, Lunteren, The Netherlands

What's The Problem?

- Consider a system with *m* customer and *n* server types
- There are two infinite streams, one of customers and one of servers
- The fraction of customers that are of type *i* equals α_i, the fraction of servers that are of type *j* equals β_j, and $\sum_{i=1}^{m} \alpha_i = \sum_{j=1}^{n} \beta_j = 1$
- A type *j* server can "match" with a type *i* customer if $j \in C_i$ (equivalently if $i \in S_j$), where C_i is the set of *servers* who can process type *i* customers (S_j is the set of *customers* who can be processed by type *j* servers)



http://stat.haifa.ac.il/~gweiss/presentations/ParallelServersFCFSMatching141108.pdf

What’s The Problem?

- Focus on problem that satisfies *complete pooling conditions*

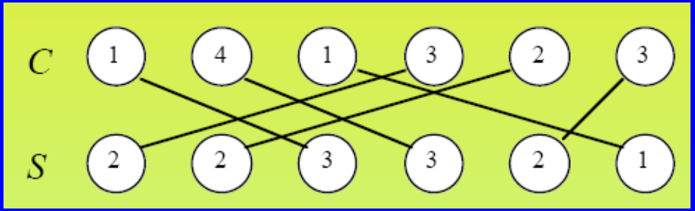
$$\alpha_i \leq \sum_{j \in C_i} \beta_j, \quad i = 1,2,\ldots,m$$

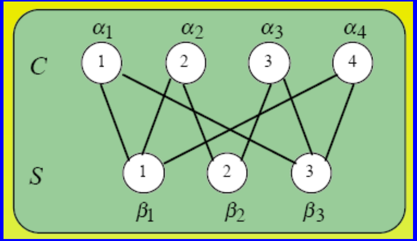
$$\beta_j \leq \sum_{i \in S_j} \alpha_i, \quad j = 1,2,\ldots,n$$

- If first condition not true, then not possible for servers to process all type *i* customers
- If second condition not true, type *j* server would be starved

What's The Problem?

- Each server in sequence is matched with the "longest waiting" feasible customer (that is, the first as yet unmatched customer in sequence eligible for service from that server, aka FCFS matching)

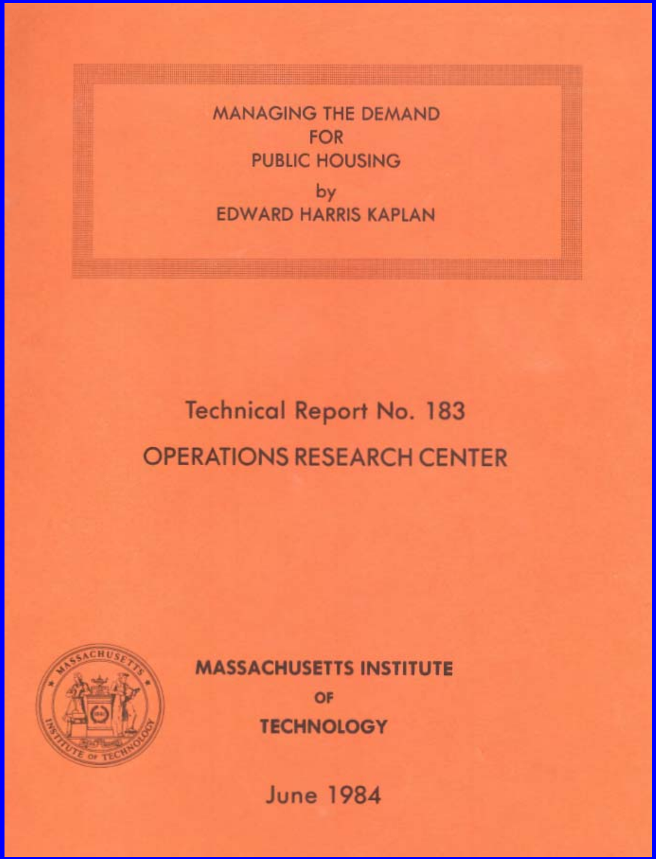




- Question: what is f_{ij}, the fraction of all matches that pair type i customers with type j servers?
- Easy question to pose; very difficult to solve in general!

http://stat.haifa.ac.il/~gweiss/presentations/ParallelServersFCFSMatching141108.pdf

Where Did This Problem First Come From?


MANAGING THE DEMAND
FOR
PUBLIC HOUSING
by
EDWARD HARRIS KAPLAN

Technical Report No. 183
OPERATIONS RESEARCH CENTER

MASSACHUSETTS INSTITUTE
OF
TECHNOLOGY

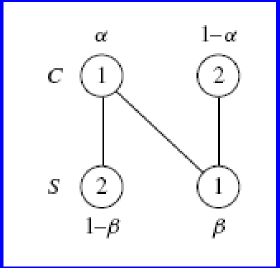
June 1984

- Boston (Massachusetts, USA) Housing Authority allowed new applicants to state housing project preferences as: willing to live only in one specific project; willing to live in one out of a specific pair of projects; willing to live in one out of a specific triple of projects; or willing to live anywhere
- The different project preferences correspond to customer types, while the different housing projects correspond to server types
- What are the matching rates?

Prior Work (More Than Matching Rates)

- Kaplan (1984) approached public housing problem as queueing using simulation
- Green (1985) solved problem with two customer and two server types (general vs limited use) using matrix geometric methods (these days known as the “N” model)

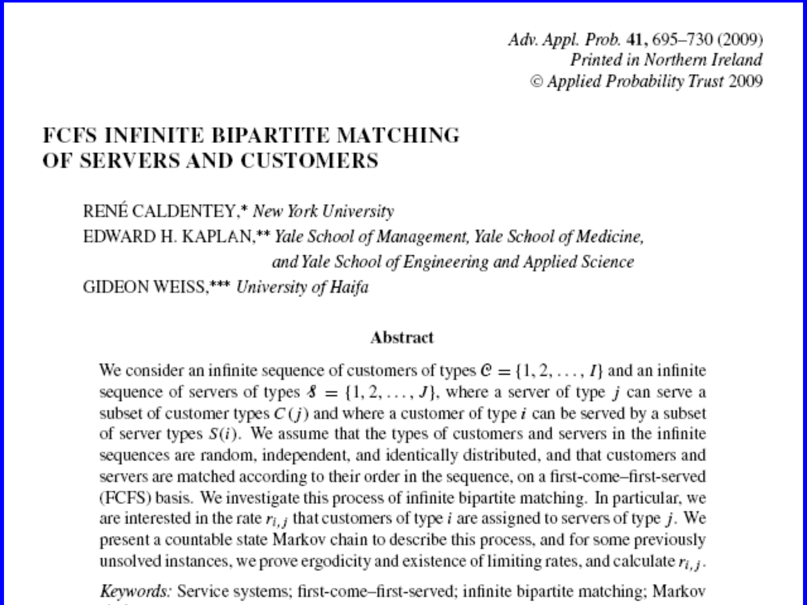
(Caldentey, Kaplan and Weiss, 2009)



- Caldentey and Kaplan (2002) introduced infinite matching approach
- Talreja and Whitt (2007) focused on fluid queueing models

Prior Work (More Than Matching Rates)

- Caldentey, Kaplan and Weiss (2009) solved special cases using Markov chains (“N”, “W”, “almost complete” graphs), and showed matching conjectures that sometimes work, sometimes not ☺



Adv. Appl. Prob. **41**, 695–730 (2009)
Printed in Northern Ireland
© *Applied Probability Trust* 2009

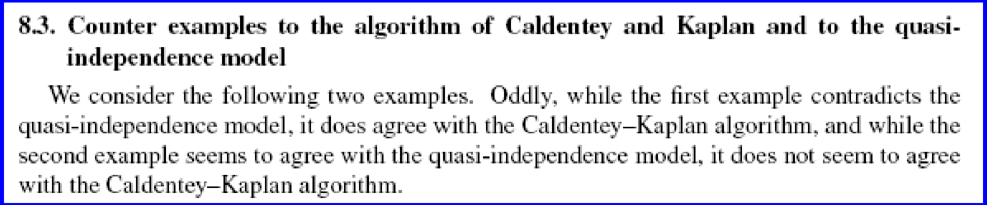
FCFS INFINITE BIPARTITE MATCHING OF SERVERS AND CUSTOMERS

RENÉ CALDENTEY,* *New York University*
EDWARD H. KAPLAN,** *Yale School of Management, Yale School of Medicine, and Yale School of Engineering and Applied Science*
GIDEON WEISS,*** *University of Haifa*

Abstract

We consider an infinite sequence of customers of types $\mathcal{C} = \{1, 2, \ldots, I\}$ and an infinite sequence of servers of types $\mathcal{S} = \{1, 2, \ldots, J\}$, where a server of type j can serve a subset of customer types $C(j)$ and where a customer of type i can be served by a subset of server types $S(i)$. We assume that the types of customers and servers in the infinite sequences are random, independent, and identically distributed, and that customers and servers are matched according to their order in the sequence, on a first-come–first-served (FCFS) basis. We investigate this process of infinite bipartite matching. In particular, we are interested in the rate $r_{i,j}$ that customers of type i are assigned to servers of type j. We present a countable state Markov chain to describe this process, and for some previously unsolved instances, we prove ergodicity and existence of limiting rates, and calculate $r_{i,j}$.

Keywords: Service systems; first-come–first-served; infinite bipartite matching; Markov



8.3. Counter examples to the algorithm of Caldentey and Kaplan and to the quasi-independence model

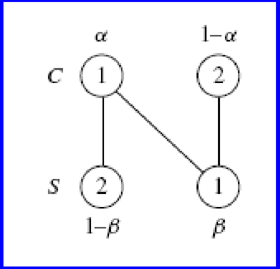
We consider the following two examples. Oddly, while the first example contradicts the quasi-independence model, it does agree with the Caldentey–Kaplan algorithm, and while the second example seems to agree with the quasi-independence model, it does not seem to agree with the Caldentey–Kaplan algorithm.

Two Easy Cases For Matching Rates

- Suppose all server types are feasible for all customer types

$$f_{ij} = \alpha_i \beta_j$$

- Suppose compatibility graph is a tree, e.g.



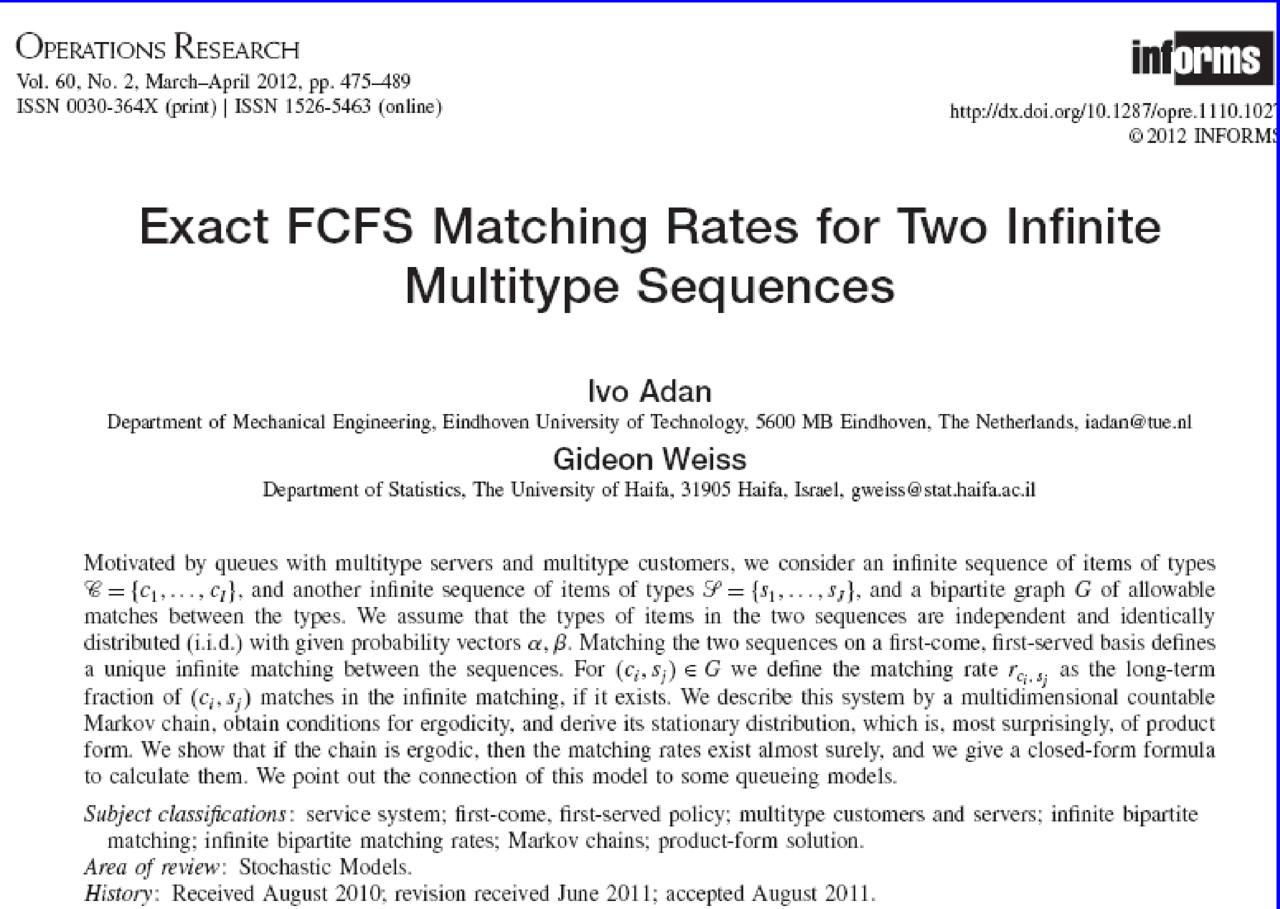
- Can just solve using flow conservation by inspection

$$f_{12} = 1 - \beta$$

$$f_{21} = 1 - \alpha$$

$$f_{11} = 1 - (1 - \beta) - (1 - \alpha) = \alpha + \beta - 1$$

Ivo Adan and Gideon Weiss Solve The Problem!



Operations Research
Vol. 60, No. 2, March–April 2012, pp. 475–489
ISSN 0030-364X (print) | ISSN 1526-5463 (online)

http://dx.doi.org/10.1287/opre.1110.102
© 2012 INFORMS

Exact FCFS Matching Rates for Two Infinite Multitype Sequences

Ivo Adan
Department of Mechanical Engineering, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands, iadan@tue.nl

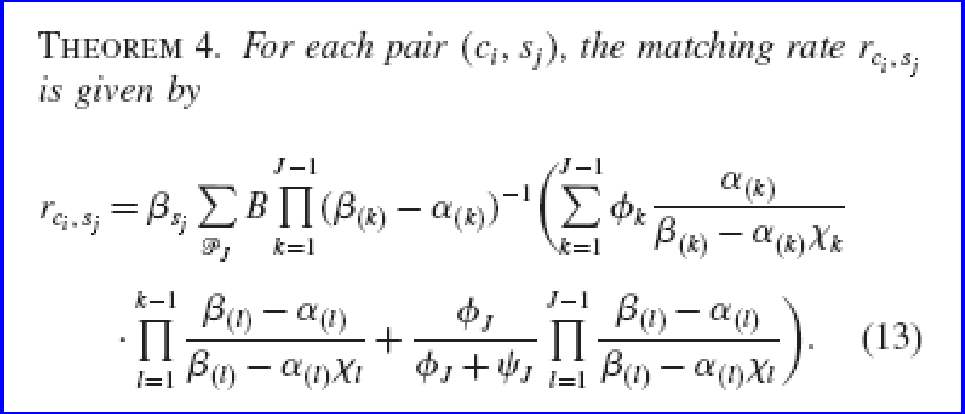
Gideon Weiss
Department of Statistics, The University of Haifa, 31905 Haifa, Israel, gweiss@stat.haifa.ac.il

Motivated by queues with multitype servers and multitype customers, we consider an infinite sequence of items of types $\mathscr{C} = \{c_1, \dots, c_I\}$, and another infinite sequence of items of types $\mathscr{S} = \{s_1, \dots, s_J\}$, and a bipartite graph G of allowable matches between the types. We assume that the types of items in the two sequences are independent and identically distributed (i.i.d.) with given probability vectors α, β. Matching the two sequences on a first-come, first-served basis defines a unique infinite matching between the sequences. For $(c_i, s_j) \in G$ we define the matching rate r_{c_i, s_j} as the long-term fraction of (c_i, s_j) matches in the infinite matching, if it exists. We describe this system by a multidimensional countable Markov chain, obtain conditions for ergodicity, and derive its stationary distribution, which is, most surprisingly, of product form. We show that if the chain is ergodic, then the matching rates exist almost surely, and we give a closed-form formula to calculate them. We point out the connection of this model to some queueing models.

Subject classifications: service system; first-come, first-served policy; multitype customers and servers; infinite bipartite matching; infinite bipartite matching rates; Markov chains; product-form solution.
Area of review: Stochastic Models.
History: Received August 2010; revision received June 2011; accepted August 2011.

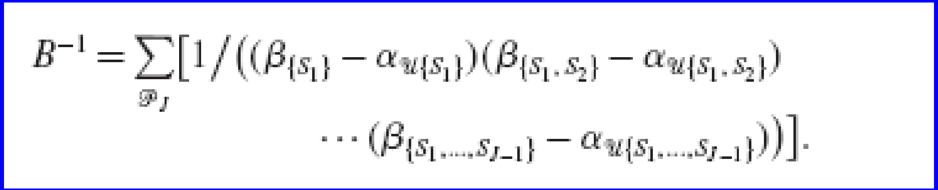
- Beautiful paper that formulates a new Markov chain, obtains its stationary distribution, and uses to obtain explicit exact formula for FCFS matching rates for general graphs

So Why Work on a Solved Problem?

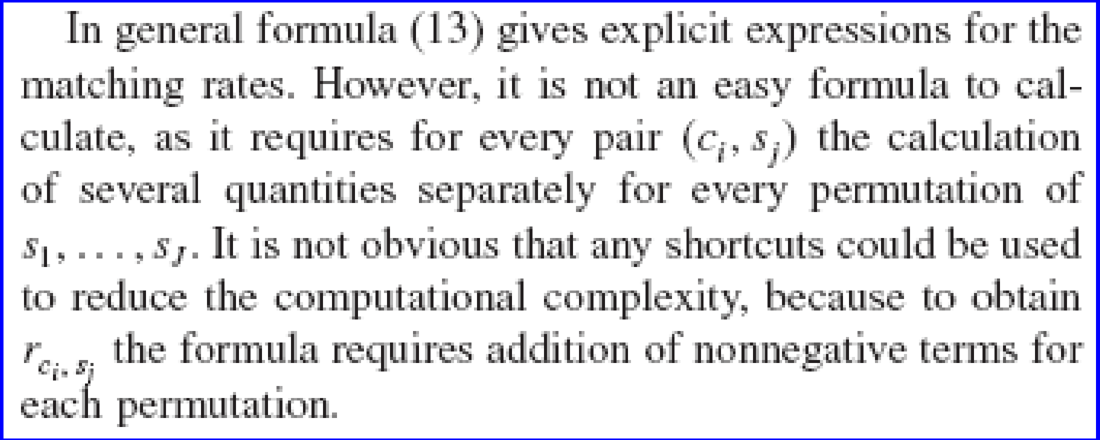


THEOREM 4. *For each pair (c_i, s_j), the matching rate r_{c_i,s_j} is given by*

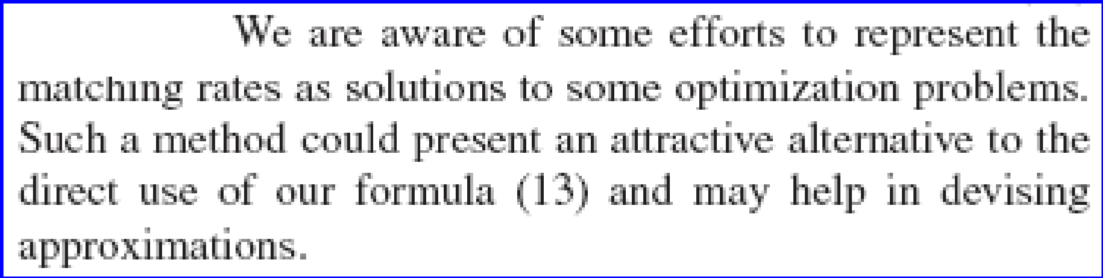
$$r_{c_i,s_j} = \beta_{s_j} \sum_{\mathcal{P}_J} B \prod_{k=1}^{J-1} (\beta_{(k)} - \alpha_{(k)})^{-1} \left(\sum_{k=1}^{J-1} \phi_k \frac{\alpha_{(k)}}{\beta_{(k)} - \alpha_{(k)}\chi_k} \cdot \prod_{l=1}^{k-1} \frac{\beta_{(l)} - \alpha_{(l)}}{\beta_{(l)} - \alpha_{(l)}\chi_l} + \frac{\phi_J}{\phi_J + \psi_J} \prod_{l=1}^{J-1} \frac{\beta_{(l)} - \alpha_{(l)}}{\beta_{(l)} - \alpha_{(l)}\chi_l} \right). \quad (13)$$



$$B^{-1} = \sum_{\mathcal{P}_J} [1/((\beta_{\{s_1\}} - \alpha_{\mathcal{U}\{s_1\}})(\beta_{\{s_1,s_2\}} - \alpha_{\mathcal{U}\{s_1,s_2\}}) \cdots (\beta_{\{s_1,\dots,s_{J-1}\}} - \alpha_{\mathcal{U}\{s_1,\dots,s_{J-1}\}}))].$$



In general formula (13) gives explicit expressions for the matching rates. However, it is not an easy formula to calculate, as it requires for every pair (c_i, s_j) the calculation of several quantities separately for every permutation of $s_1, \dots, s_J$. It is not obvious that any shortcuts could be used to reduce the computational complexity, because to obtain r_{c_i,s_j} the formula requires addition of nonnegative terms for each permutation.



We are aware of some efforts to represent the matching rates as solutions to some optimization problems. Such a method could present an attractive alternative to the direct use of our formula (13) and may help in devising approximations.

So, here we are!!

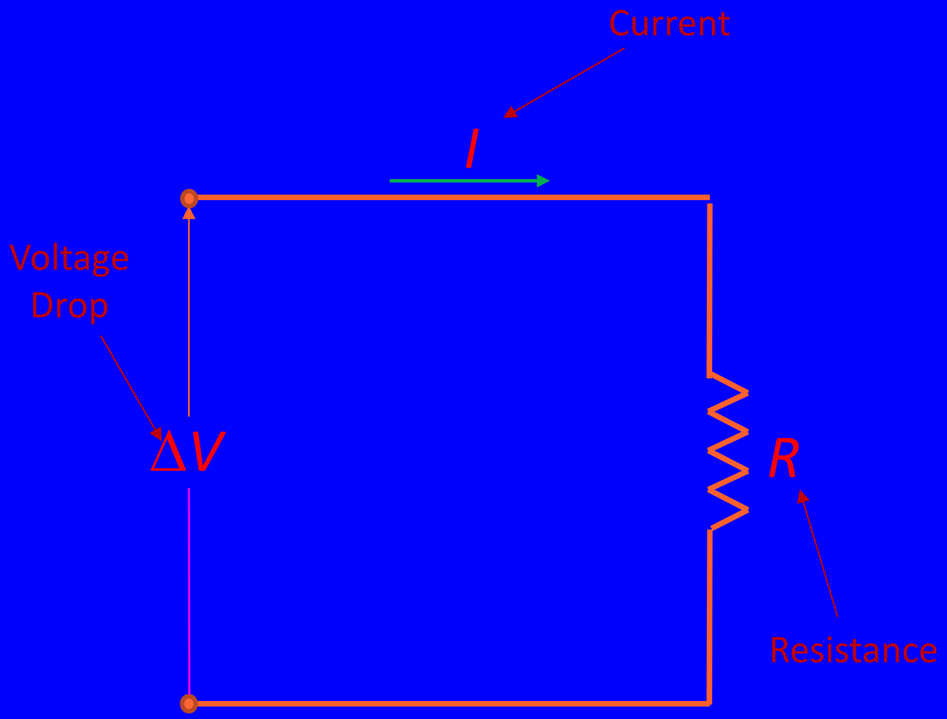
Physical Flows

- Problem can be imagined as sending water over a pipe network linking sources (customer types) with given inflows to sinks (server types) with given outflows
- Unlike usual min cost or max flow problems, the question is given the inflows and outflows, how does water distribute itself over the pipe network?
- Physical principles: Bernoulli equation, Kirchhoff’s laws
- Classic reference: Analysis of flow in networks of conduits or conductors (Hardy Cross, U Illinois Bulletin 34(22), 1936)

Physical Flows

- Problem can be imagined as sending electricity over a circuit linking various voltage sources (customer types) with given current inflows to voltage sinks (server types) with given current outflows
- Question: how does the current distribute itself over the network?
- Physical principle: Ohm’s Law
- Nice thing about Ohm’s Law as we will see: linearity!

Ohm’s Law



$$I = \frac{\Delta V}{R}$$

Ohm's Law On A Bipartite Graph

- *m* sources (customer types) and *n* sinks (server types)
- Current inflow at source *i* equals α_i, current outflow at sink *j* equals β_j, and $\sum_{i=1}^{m} \alpha_i = \sum_{j=1}^{n} \beta_j = 1$
- A type *j* server can "match" with a type *i* customer if $j \in C_i$ (equivalently if $i \in S_j$), where C_i is the set of *servers* who can process type *i* customers (S_j is the set of *customers* who can be processed by type *j* servers)
- Voltage at source *i* equals V_i, voltage at sink *j* equals W_j
- Take resistance on link $(i, j) = (\alpha_i \beta_j)^{-1}$

Ohm's Law On A Bipartite Graph

- Flows f_{ij} must all satisfy Ohm's Law $I = \frac{\Delta V}{R}$
- On link (i, j) voltage drop equals $V_i - W_j$ while resistance equals $(\alpha_i \beta_j)^{-1}$, thus

$$f_{ij} = \alpha_i \beta_j (V_i - W_j)$$

- Flows must also satisfy given current inflows and outflows

$$\sum_{j \in C_i} f_{ij} = \sum_{j \in C_i} \alpha_i \beta_j (V_i - W_j) = \alpha_i, \quad i = 1, 2, \ldots, m$$

$$\sum_{i \in S_j} f_{ij} = \sum_{i \in S_j} \alpha_i \beta_j (V_i - W_j) = \beta_j, \quad j = 1, 2, \ldots, n$$

Ohm's Law On Bipartite Graph

- So we need to solve

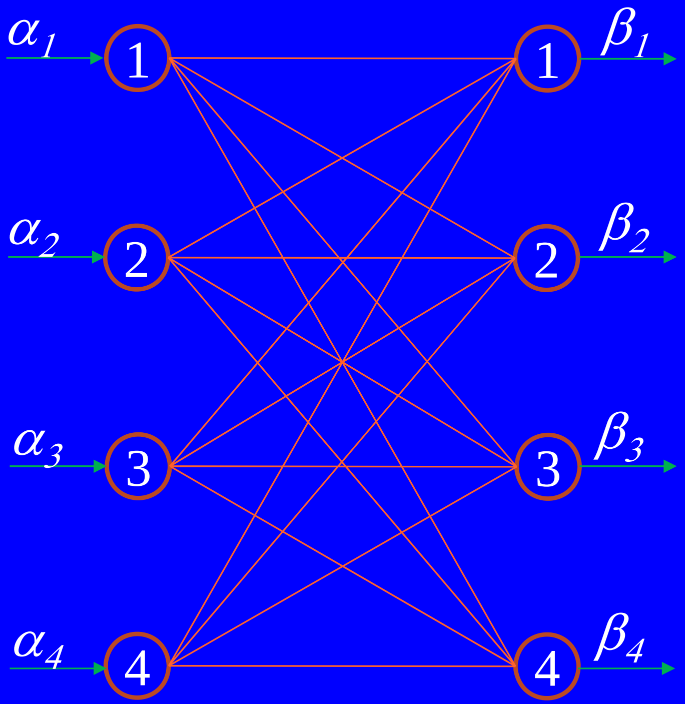
$$\sum_{j\in C_i} f_{ij} = \sum_{j\in C_i} \alpha_i\beta_j(V_i - W_j) = \alpha_i, \quad i = 1,2,\ldots,m$$

$$\sum_{i\in S_j} f_{ij} = \sum_{i\in S_j} \alpha_i\beta_j(V_i - W_j) = \beta_j, \quad j = 1,2,\ldots,n$$

for the unknown voltages V_i and W_j

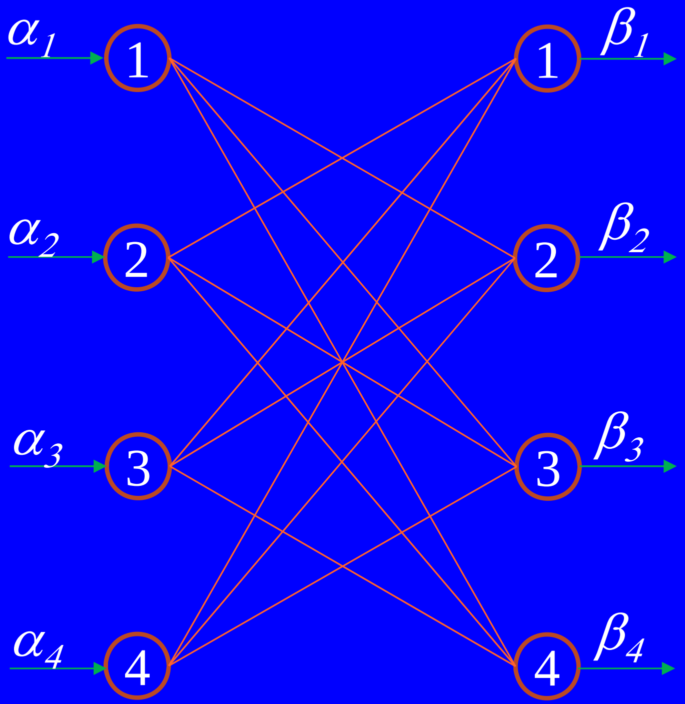
- Note that system above only defined to additive constant (can add constant c to all voltages without changing flows)
- Can impose any convenient restriction to solve (e.g. $W_n = 0$)

Does It Work?



- Let's try for completely connected graph
- We know answer is $f_{ij} = \alpha_i \beta_j$
- Ohm's Law says $f_{ij} = \alpha_i \beta_j (V_i - W_j)$
- Set $V_i = 1$, $W_j = 0$.
- Yay!!

Almost Complete Graph (*n* – 1 graph)



- From Caldentey, Kaplan and Weiss (2009) as well as Adan and Weiss (2012), we know the exact answer

$$r_{i,j} = \frac{\alpha_i \beta_j [(1-\alpha_i)(1-\beta_j) - \alpha_j \beta_i]}{(1-\alpha_i-\beta_i)(1-\alpha_j-\beta_j)} \pi_0^Z$$

where $r_{i,j} = f_{ij}$ in our notation, and π_0^Z is a normalization constant

An Almost Complete Surprise

- So for almost complete graph we know that f_{ij} =

$$r_{i,j} = \frac{\alpha_i \beta_j[(1-\alpha_i)(1-\beta_j) - \alpha_j \beta_i]}{(1-\alpha_i-\beta_i)(1-\alpha_j-\beta_j)} \pi_0^Z$$

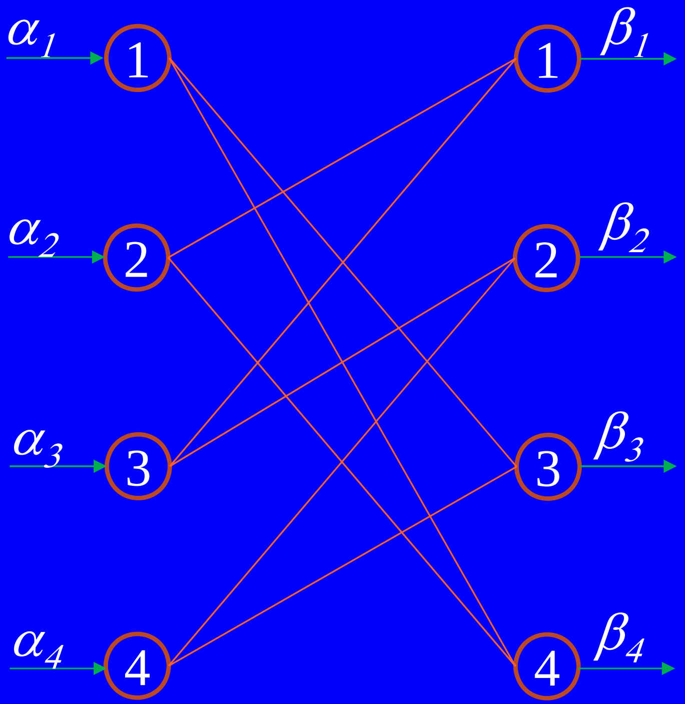
- Let

$$V_i = \frac{\beta_i}{1-\alpha_i-\beta_i} \pi_0^Z$$

$$W_j = \frac{\beta_j - 1}{1-\alpha_j-\beta_j} \pi_0^Z$$

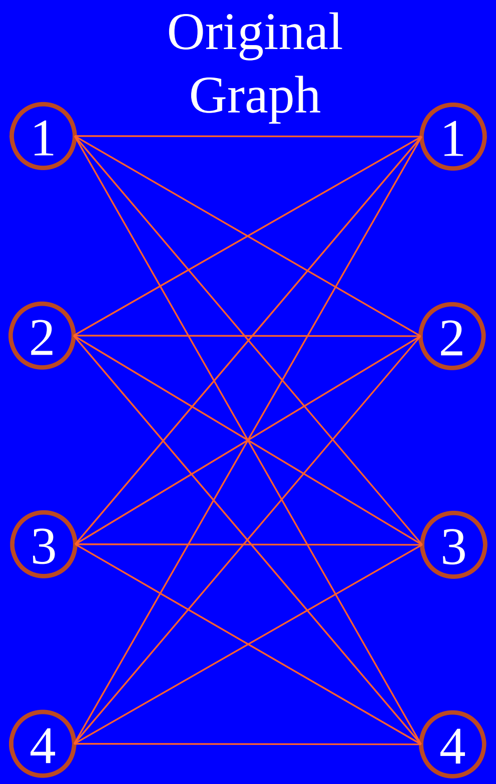
- Guess what? $f_{ij} = \alpha_i \beta_j (V_i - W_j)$ yields correct solution!
- So get correct answer directly from Ohm’s Law!

What About *n* – 2 Graph?



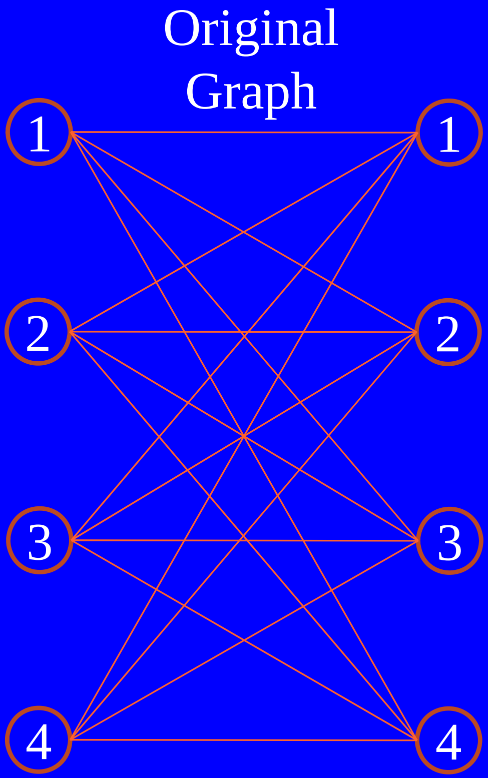
- Each customer type connected to all but two server types
- Sad news – Ohm’s Law doesn’t work directly
- But can we learn anything from studying Adan and Weiss formula for this case?

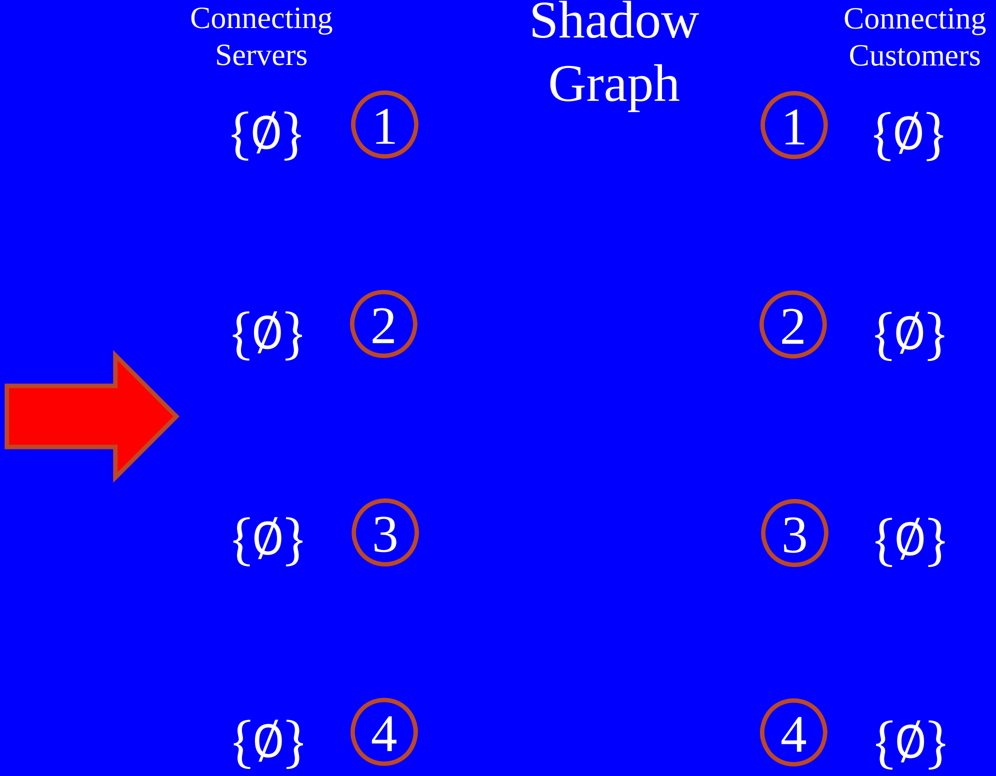
Introducing: The Shadow Graph



- Start with a completely connected graph
- Shadow graph is formed by keeping same source and sink nodes but inserting the *complement* of the links in the original graph
- Shadow graph of completely connected graph will be empty (no links)

The Shadow Graph

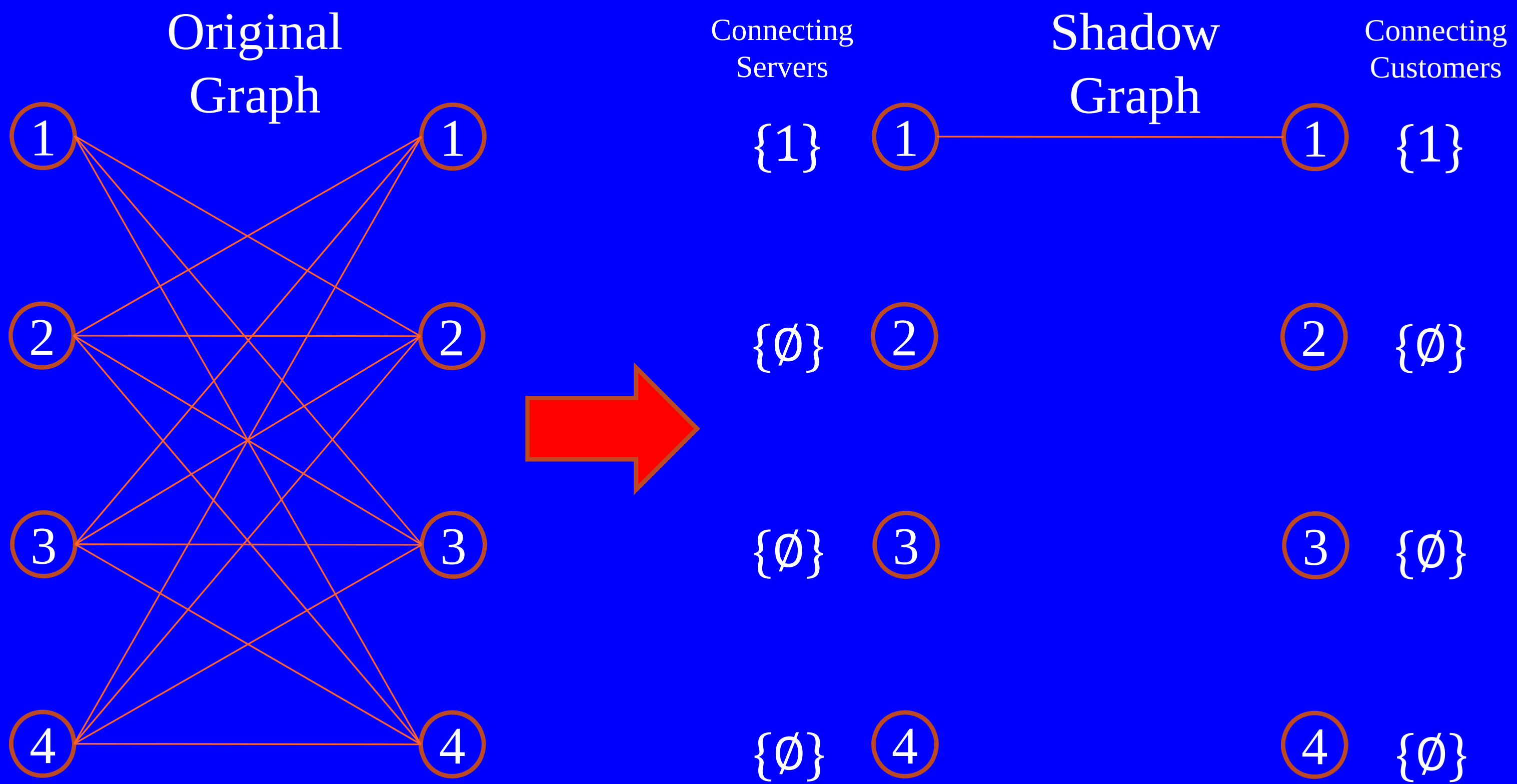




The Shadow Graph

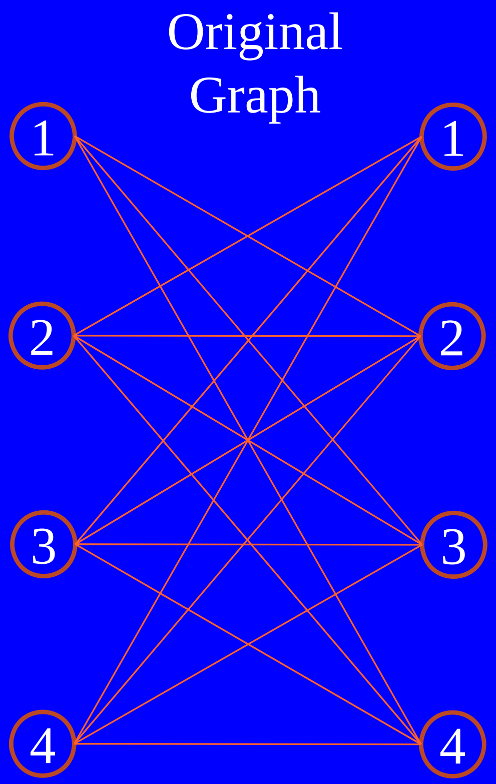
- Now, as we remove links from the original graph, we add those links to the shadow graph
- For each customer node in the shadow graph, we list all connecting servers
- Similarly, for each server node in the shadow graph, we list all connecting customers
- Then, for each *link* in the shadow graph, we form all combinations of customers who can be processed by the link’s server type, and servers who can process the link’s customer type
- If we find a customer/server pair that is *not* represented by a link in the shadow graph, that becomes a *target link* for excess flow in the original graph from the *hidden link* in the shadow graph

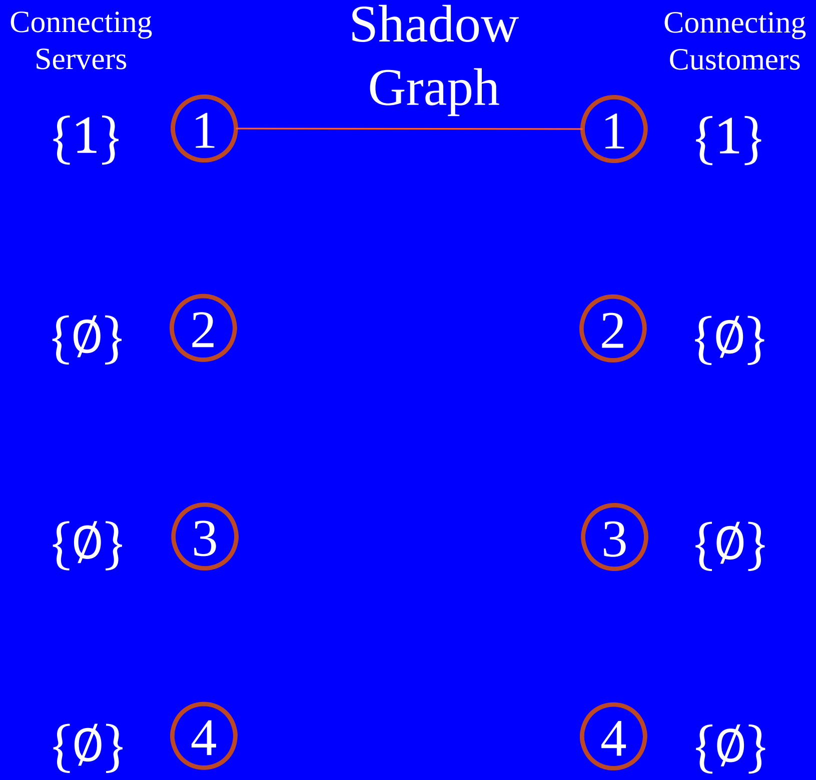
The Shadow Graph



Shadow Link	Combinations
1,1	1,1

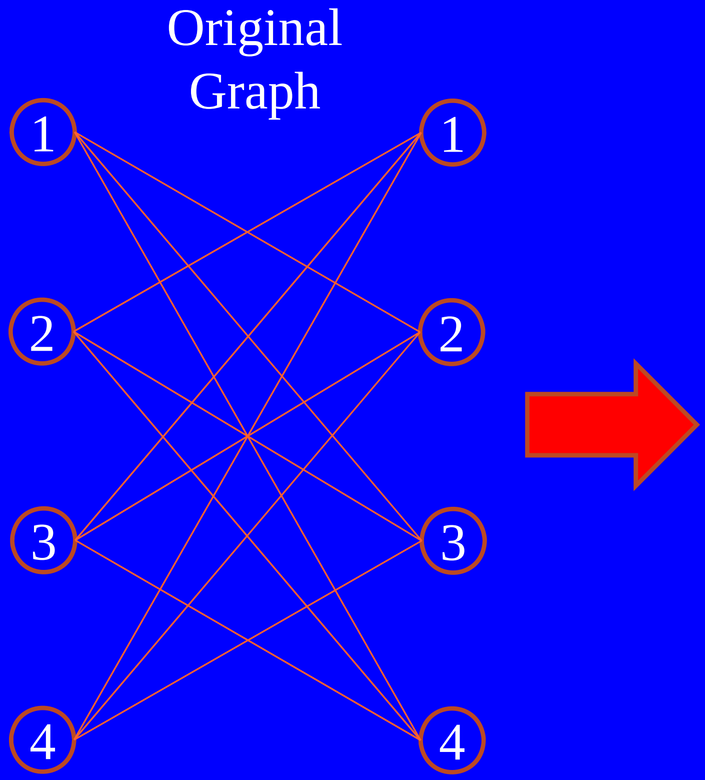
The Shadow Graph

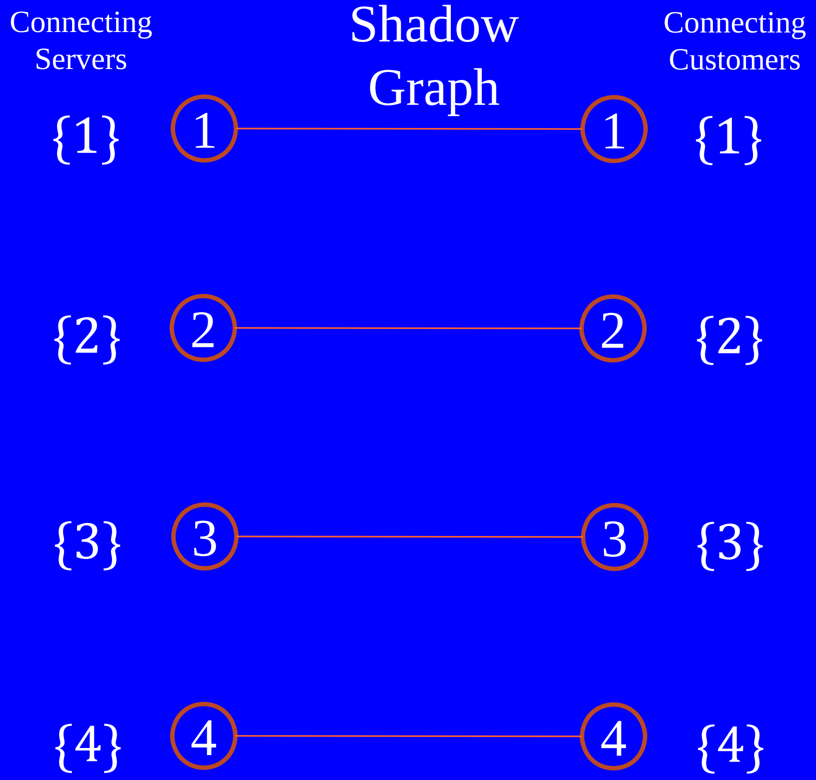




Shadow Link	Combinations
1,1	~~1,1~~

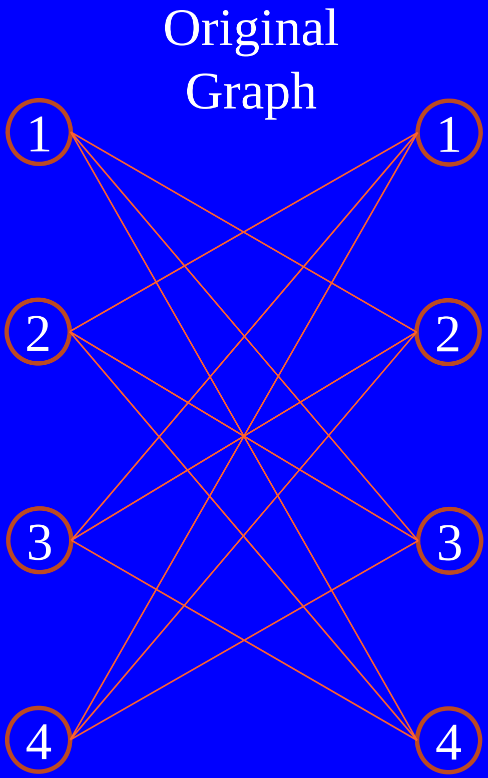
The Shadow Graph

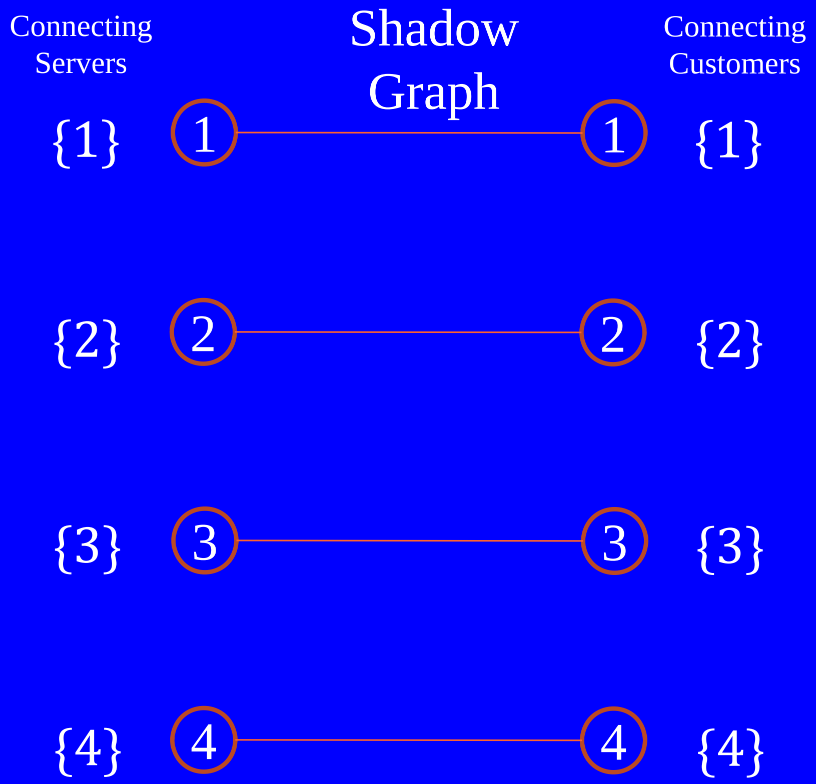




Shadow Link	Combinations
1,1	1,1
2,2	2,2
3,3	3,3
4,4	4,4

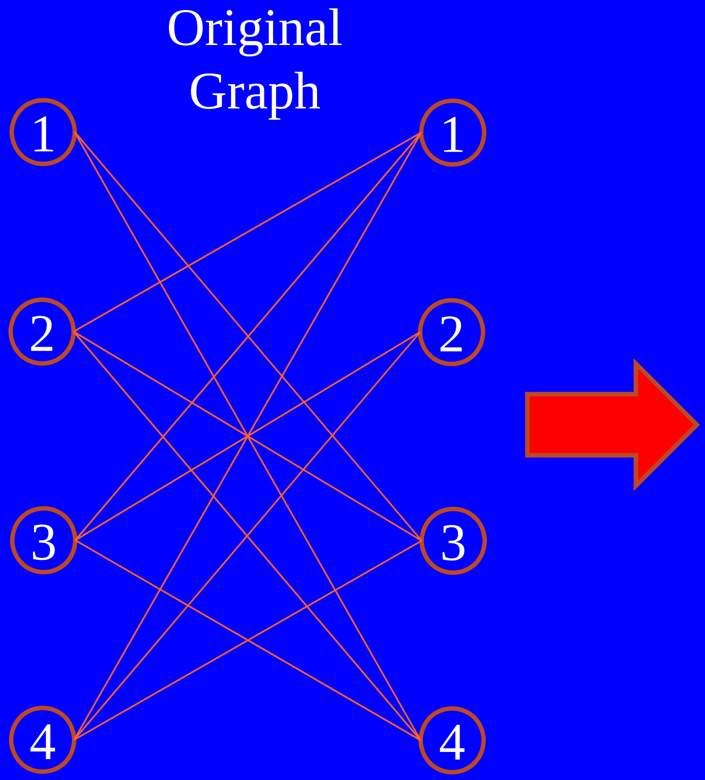
The Shadow Graph

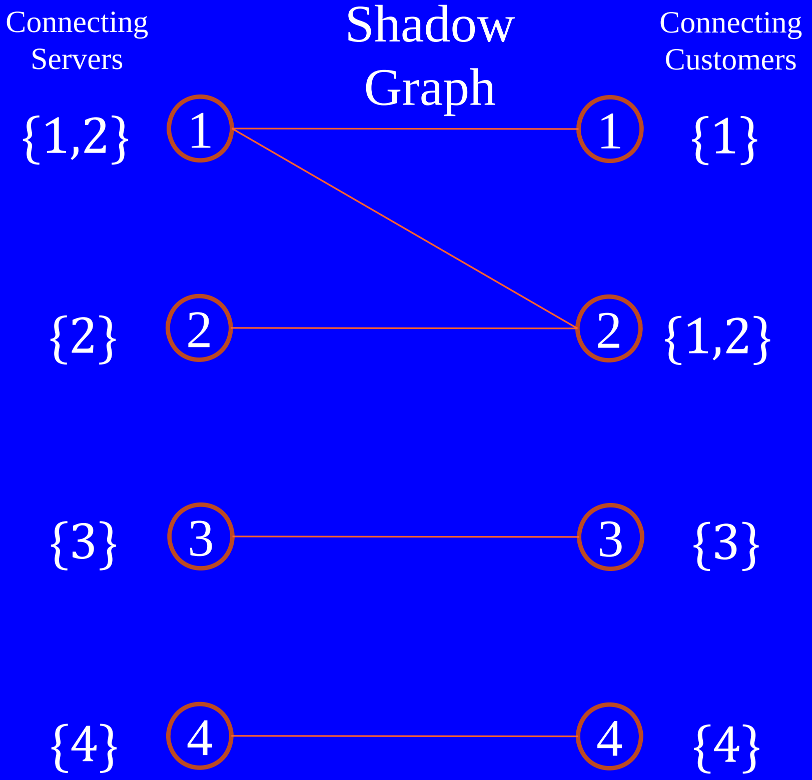




Shadow Link	Combinations
1,1	~~1,1~~
2,2	~~2,2~~
3,3	~~3,3~~
4,4	~~4,4~~

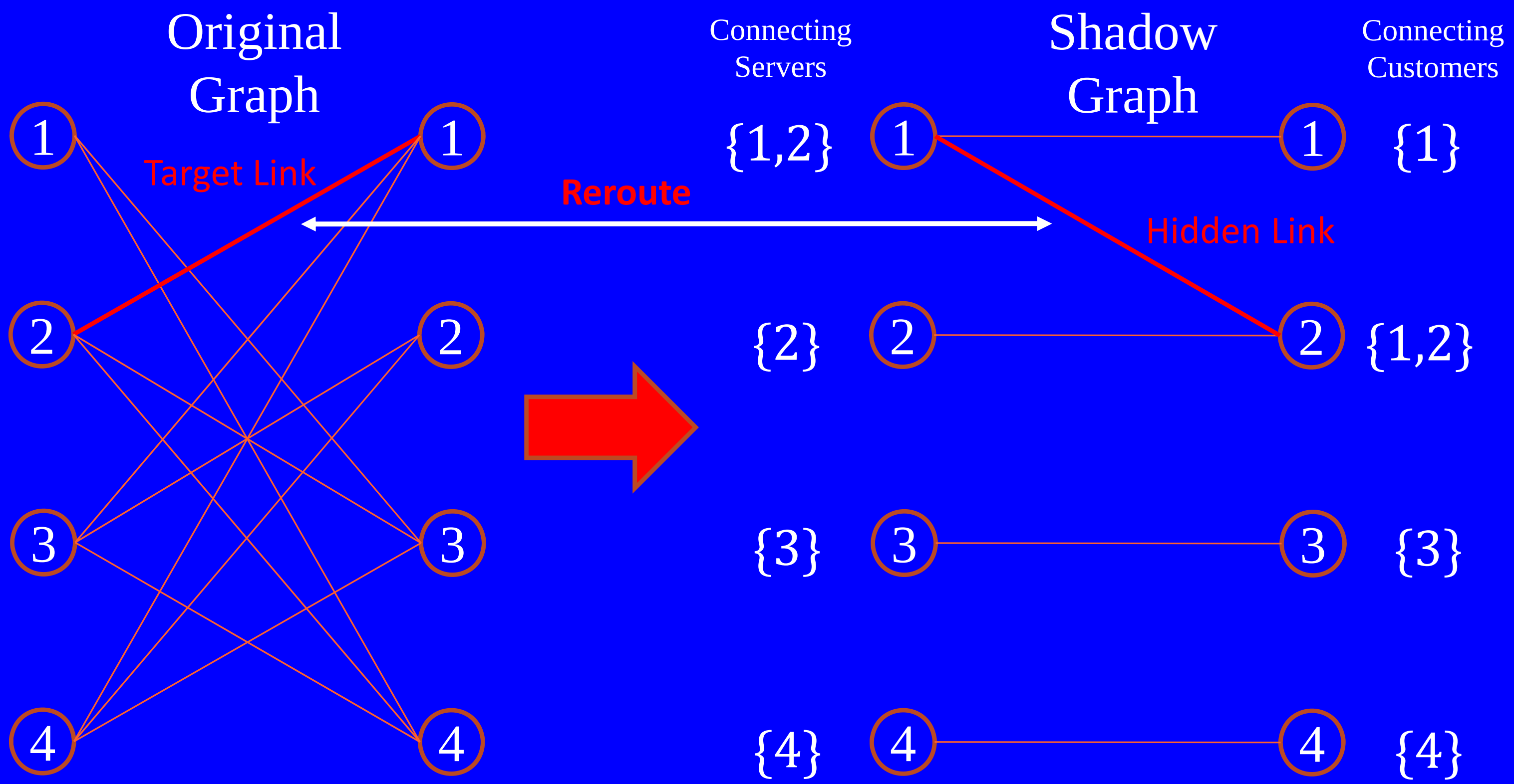
The Shadow Graph





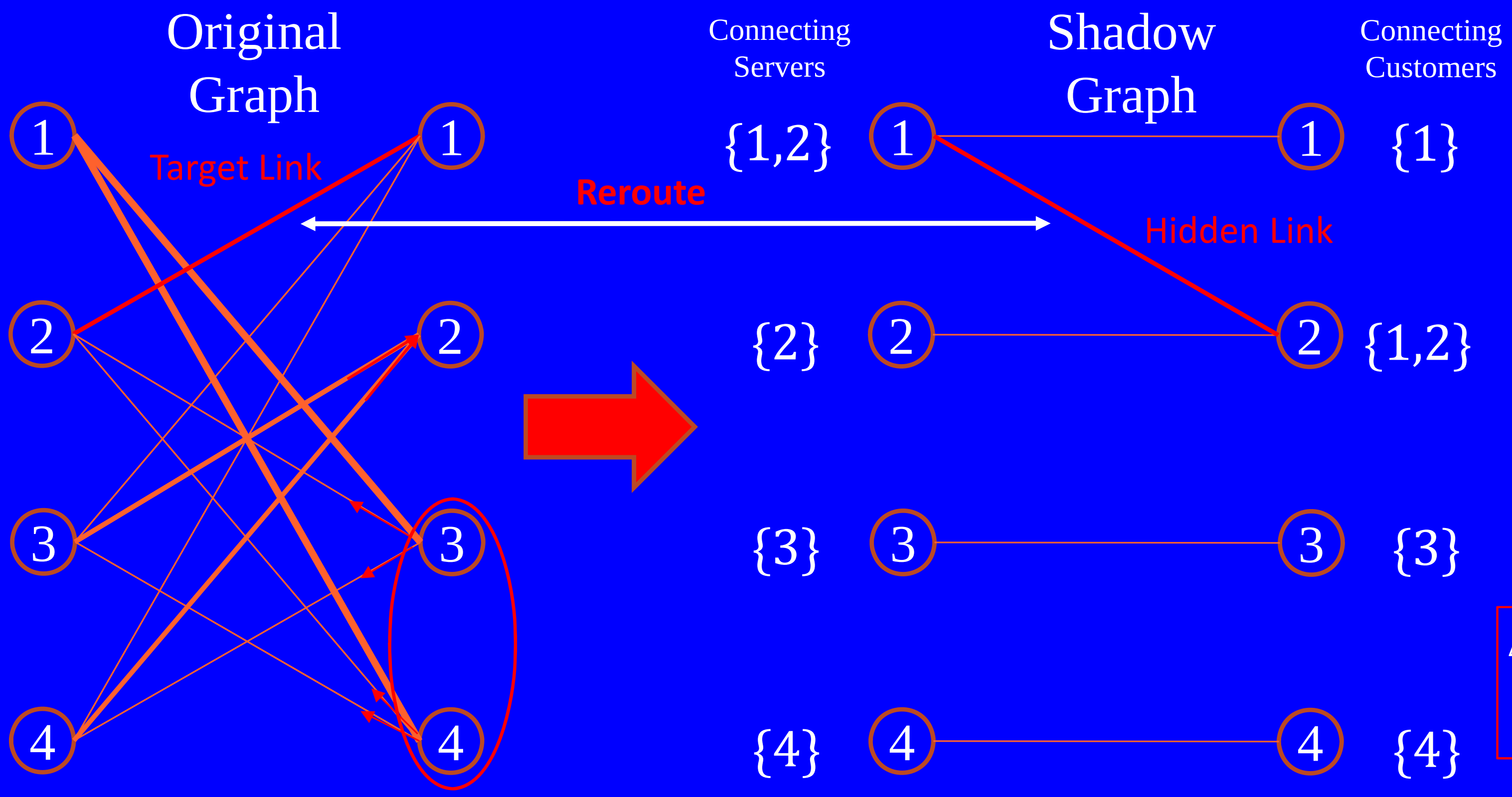
Shadow Link	Combinations
1,1	1,1 ; 1,2
1,2	1,1 ; 1,2 2,1 ; 2,2
2,2	2,2
3,3	3,3
4,4	4,4

The Shadow Graph



Shadow Link	Combinations
1,1	~~1,1~~ ; ~~1,2~~
1,2	~~1,1~~ ; ~~1,2~~ 2,1 ; ~~2,2~~
2,2	~~2,2~~
3,3	~~3,3~~
4,4	~~4,4~~

The Shadow Graph



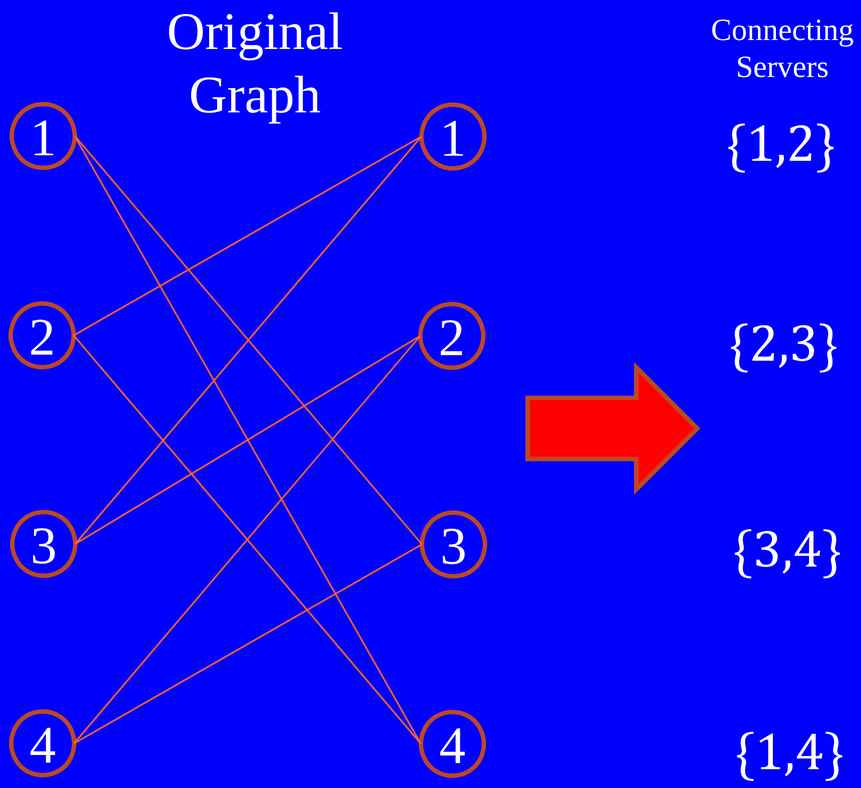
Shadow Link	Combinations
1,1	~~1,1~~ ; ~~1,2~~
1,2	~~1,1~~ ; ~~1,2~~ 2,1 ; ~~2,2~~
2,2	~~2,2~~
3,3	~~3,3~~
4,4	~~4,4~~

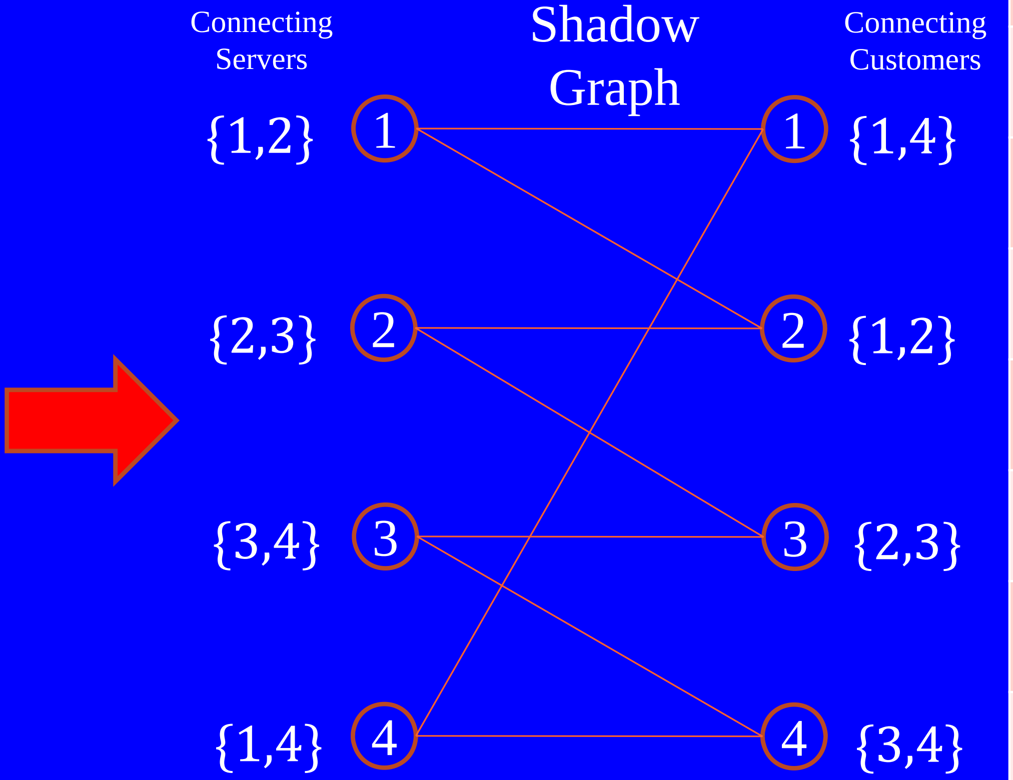
Add to f_{21}: $k\Delta_{21}$

$$\Delta_{21} = \frac{\alpha_1}{\beta_3 + \beta_4 - \alpha_1} \times \frac{\beta_2}{\alpha_3 + \alpha_4 - \beta_2}$$

$$0 < k < 1$$

The Shadow Graph





Shadow Link	Combinations
1,1	1,1 ; 1,2 4,1 ; 4,2
1,2	1,1 ; 1,2 2,1 ; 2,2
2,2	1,2 ; 1,3 2,2 ; 2,3
2,3	2,2 ; 2,3 3,2 ; 3,3
3,3	2,3 ; 2,4 3,3 ; 3,4
3,4	3,3 ; 3,4 4,3 ; 4,4
4,4	3,1 ; 3,4 4,1 ; 4,4
4,1	1,1 ; 1,4 4,1 ; 4,4

The Shadow Graph Procedure

1. Identify the *hidden links* i', j' from the shadow graph
2. For each link *i,j* in the original graph, modify the flow equation by adding the corresponding hidden link formula:

$$k\Delta_{ij} = k\frac{\alpha_{i'}}{\left(\sum_{j\in C_{i'}}\beta_j\right) - \alpha_{i'}} \times \frac{\beta_{j'}}{\left(\sum_{i\in S_{j'}}\alpha_i\right) - \beta_{j'}}$$

3. Choose an approximate value for $0 < k < 1$. Two suggestions are:

   i. Mid point approximation $k = 0.5$

   ii. Proportional approximation $k = \min\left\{\frac{1}{\sum_{shadow}\Delta_{ij}}, 1\right\}$

4. Given the modified flow equations: $f_{ij} = \alpha_i\beta_j\left[V_i - W_j + k\Delta_{i,j}\right]$, solve for V_is and W_js.

$$\sum_{j\in C_i} f_{ij} = \sum_{j\in C_i}\alpha_i\beta_j(V_i - W_j + k\Delta_{ij}) = \alpha_i, \quad i = 1,2,\ldots,m$$

$$\sum_{i\in S_j} f_{ij} = \sum_{i\in S_j}\alpha_i\beta_j(V_i - W_j + k\Delta_{ij}) = \beta_j, \quad j = 1,2,\ldots,n$$

Results

- Given the correct value of *k* (which equals $B / \prod_{j=1}^{n} \beta_j$ where *B* is given in equation (12) of Adan and Weiss), the shadow graph procedure produces the exact solution for any *n-2* to *n* graph.
- Suppose $F_{k=0}$ and $F_{k=1}$ are the solutions of the shadow graph procedure when $k = 0$ and $k = 1$, respectively. Then, for any *n-2* to *n* graph, the exact solutions $f_{i,j}$ are contained in the intervals bounded by $F_{k=0}$ and $F_{k=1}$. Moreover, $f_{ij}s$ are linear in k.

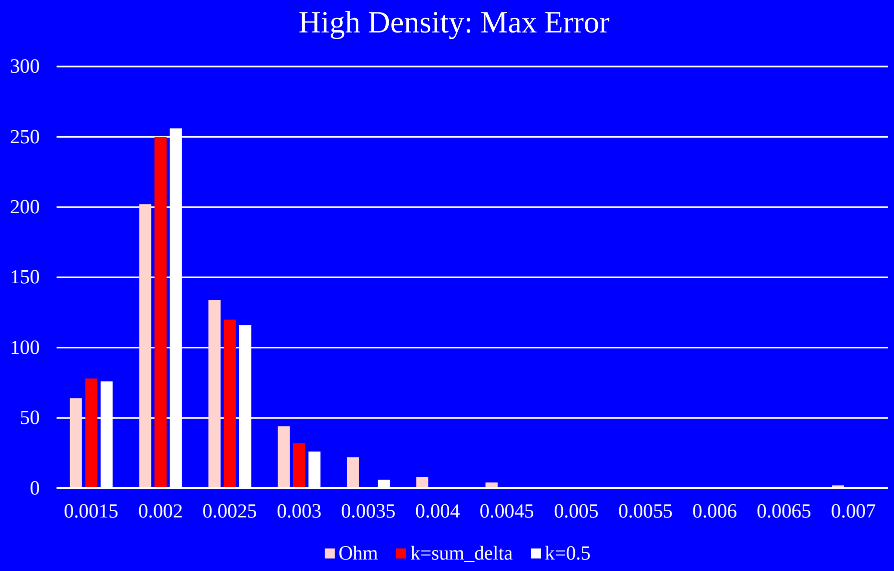
Idea For Approximation

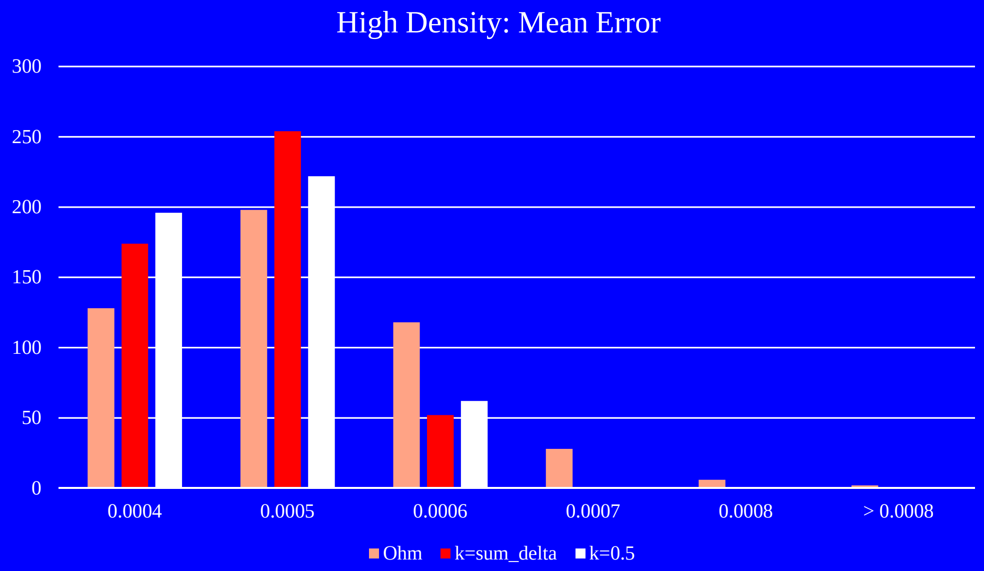
- Nice thing about shadow graph approach is that it yields exact results in principle (given *k*) for family of graphs
- Leads to easy system of linear equations to solve (same size as original Ohm’s Law)
- So, try shadow graph approach to arbitrary networks
- Simulation time

Experiments: Problem Instances

- Number of Customers = 10
- Number of Servers = Random between 7 and 10
- Connectivity Graph
  - Random
    - » High Density: Each customer is connected to at least 70% of servers
    - » Medium Density: Each customer is connected to between 40% to 70% of the servers
    - » Low Density: Each customer is connected to at most 40% the servers
- $\alpha = \{\alpha_i\}, \beta = \{\beta_i\}$: Randomly generated such that they sum up to 1
- Total number of problem instances = 1500:
  - For each density: 20 rand. con. graph $\times$ 25 rand. α, β

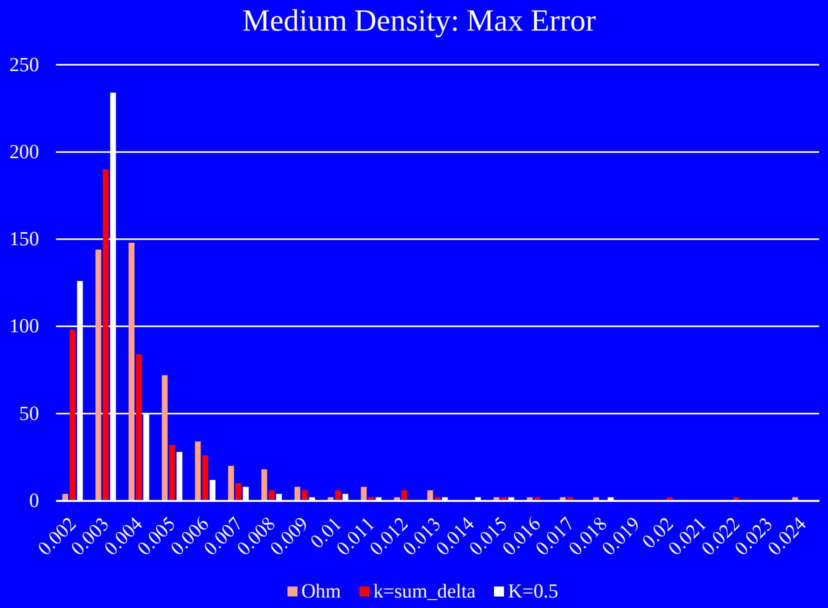
Experiments: High Density

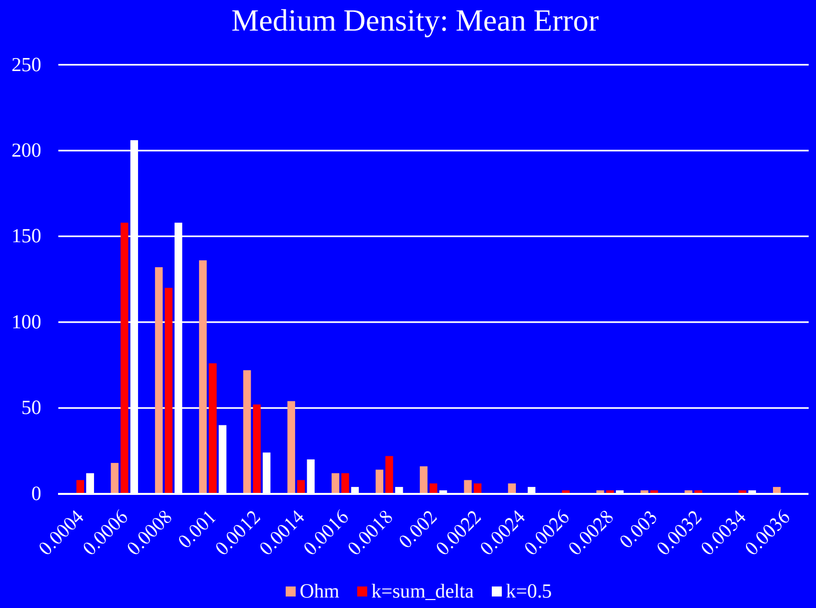




	Ohm	k=sum_delta	k=0.5
Ave. Max. Error	0.0021	0.0019	0.0019
Mean Ave. Error	0.0005	0.0005	0.0005

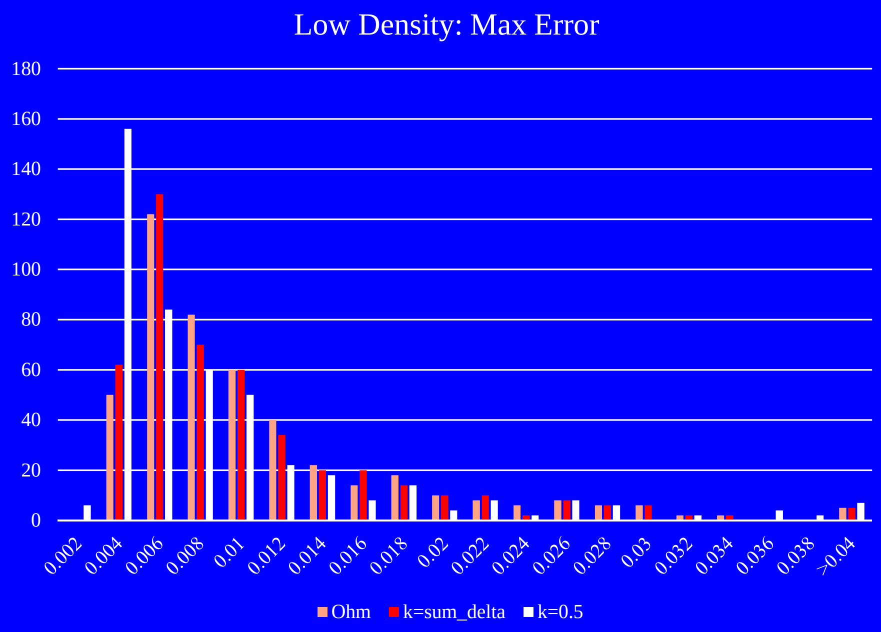
Experiments: Medium Density

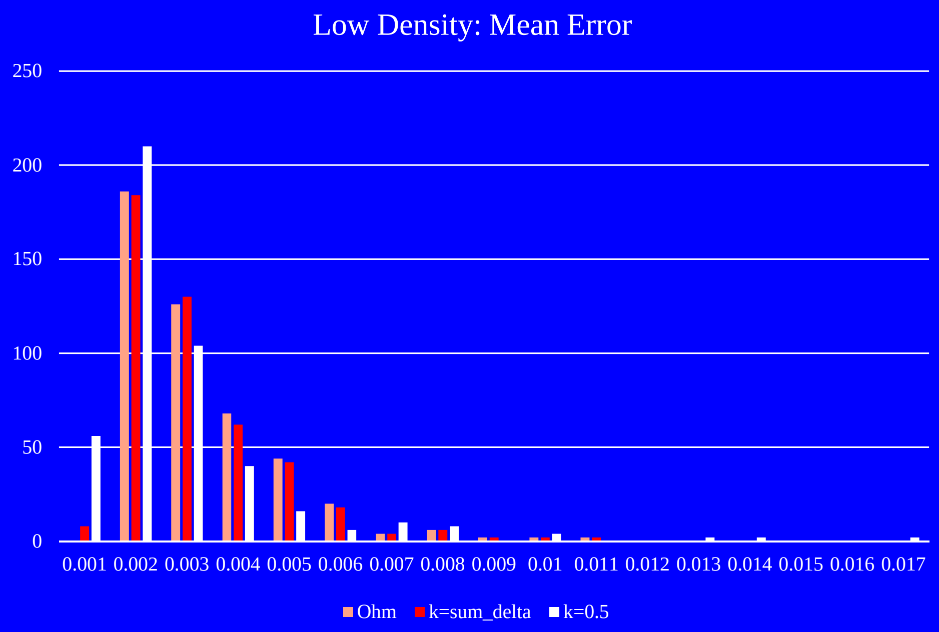




	Ohm	k=sum_delta	k=0.5
Ave. Max. Error	0.0045	0.0036	0.0030
Mean Ave. Error	0.0011	0.0009	0.0008

Experiments: Low Density

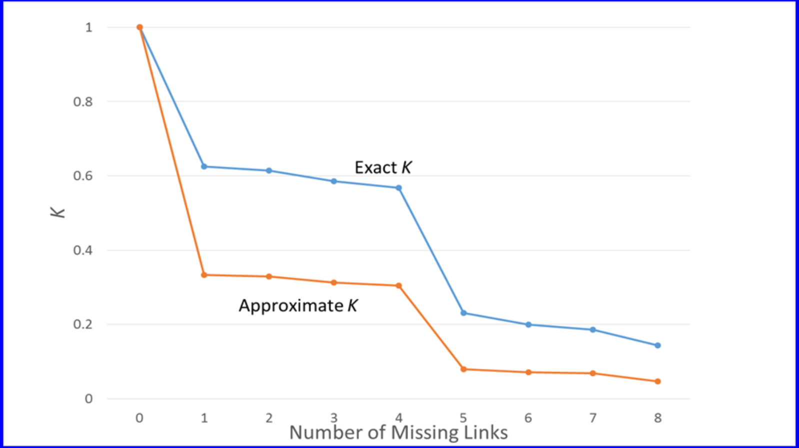


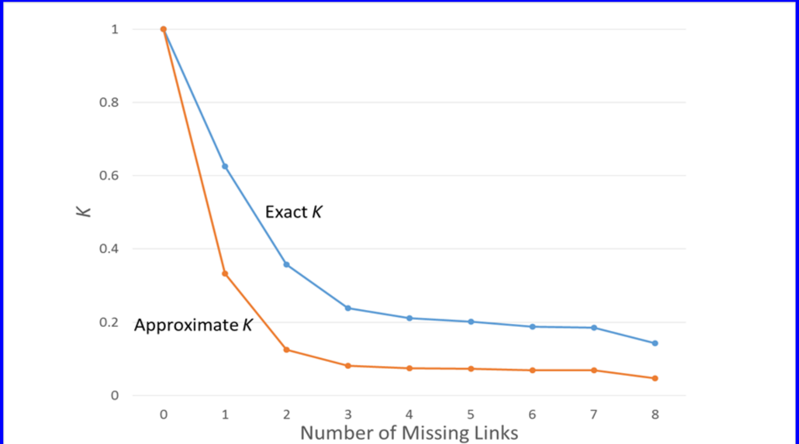


	Ohm	k=sum_delta	k=0.5
Ave. Max. Error	0.0098	0.0095	0.0084
Mean Ave. Error	0.0028	0.0027	0.0024

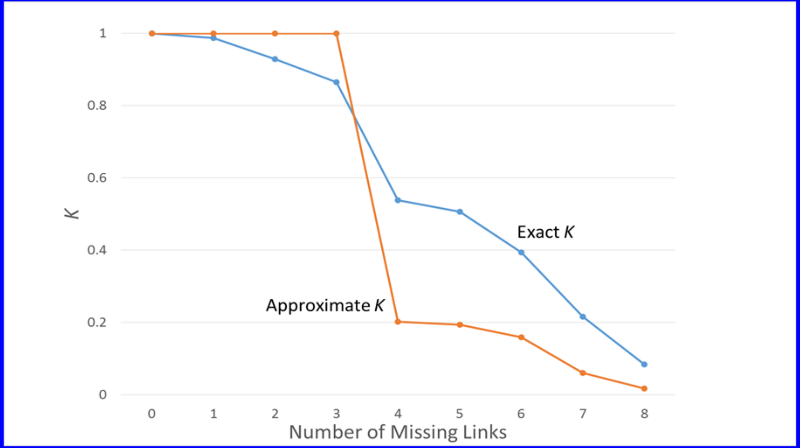
Experiments: The Behavior of *K*

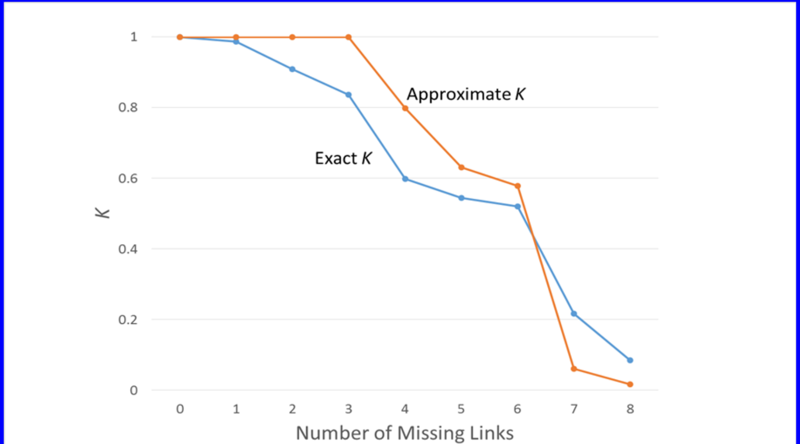
$\alpha_1 = 0.6, \alpha_2 = 0.2, \alpha_3 = 0.1, \alpha_4 = 0.1$
$\beta_1 = 0.2, \beta_2 = 0.1, \beta_3 = 0.4, \beta_4 = 0.3$





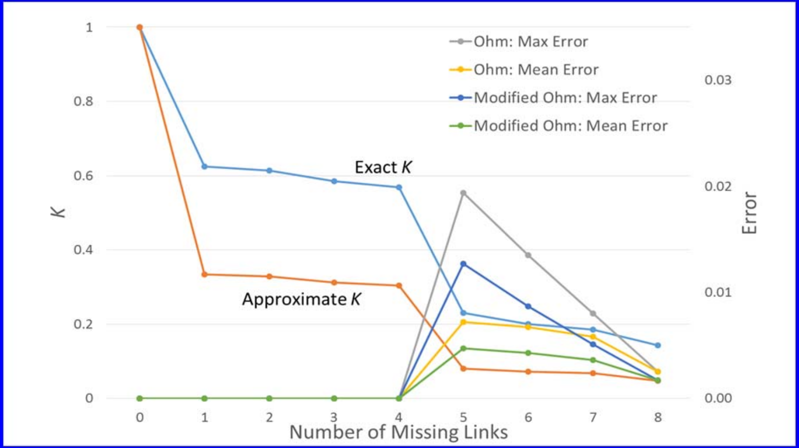
$\alpha_1 = 0.1, \alpha_2 = 0.1, \alpha_3 = 0.1, \alpha_4 = 0.7$
$\beta_1 = 0.1, \beta_2 = 0.35, \beta_3 = 0.4, \beta_4 = 0.15$

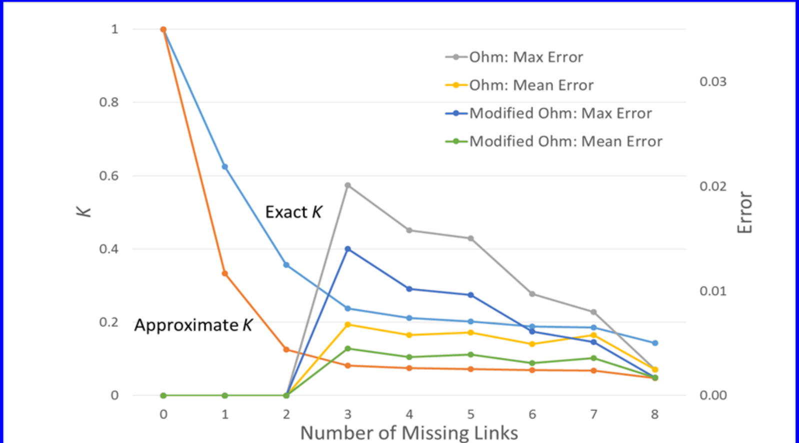




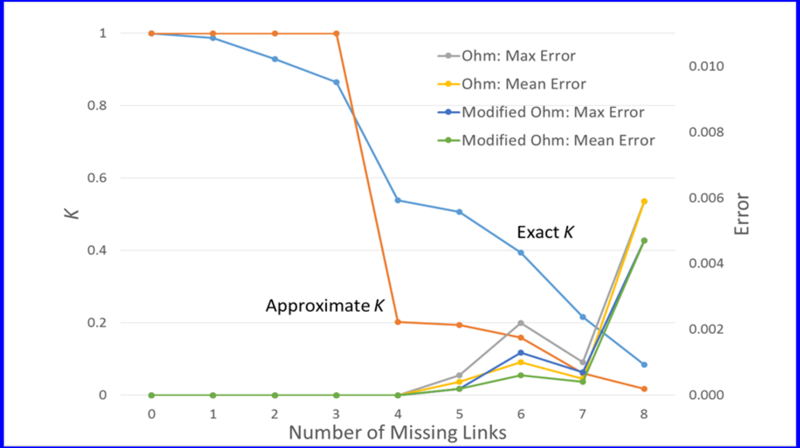
Experiments: The Behavior of *K*

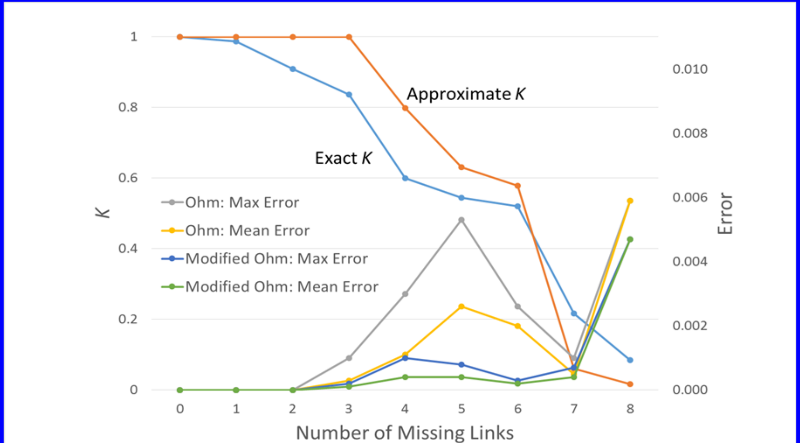
$\alpha_1 = 0.6\,,\ \alpha_2 = 0.2\,,\ \alpha_3 = 0.1\,,\ \alpha_4 = 0.1$
$\beta_1 = 0.2\,,\ \beta_2 = 0.1\,,\ \beta_3 = 0.4\,,\ \beta_4 = 0.3$



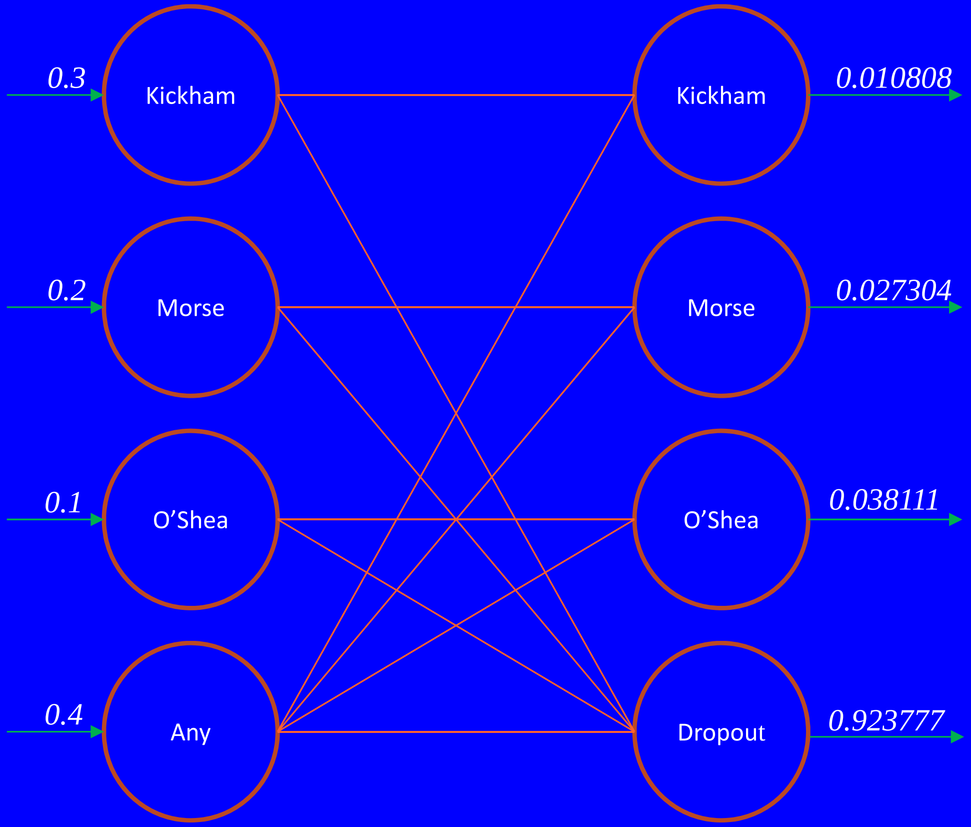


$\alpha_1 = 0.1\,,\ \alpha_2 = 0.1\,,\ \alpha_3 = 0.1\,,\ \alpha_4 = 0.7$
$\beta_1 = 0.1\,,\ \beta_2 = 0.35\,,\ \beta_3 = 0.4\,,\ \beta_4 = 0.15$



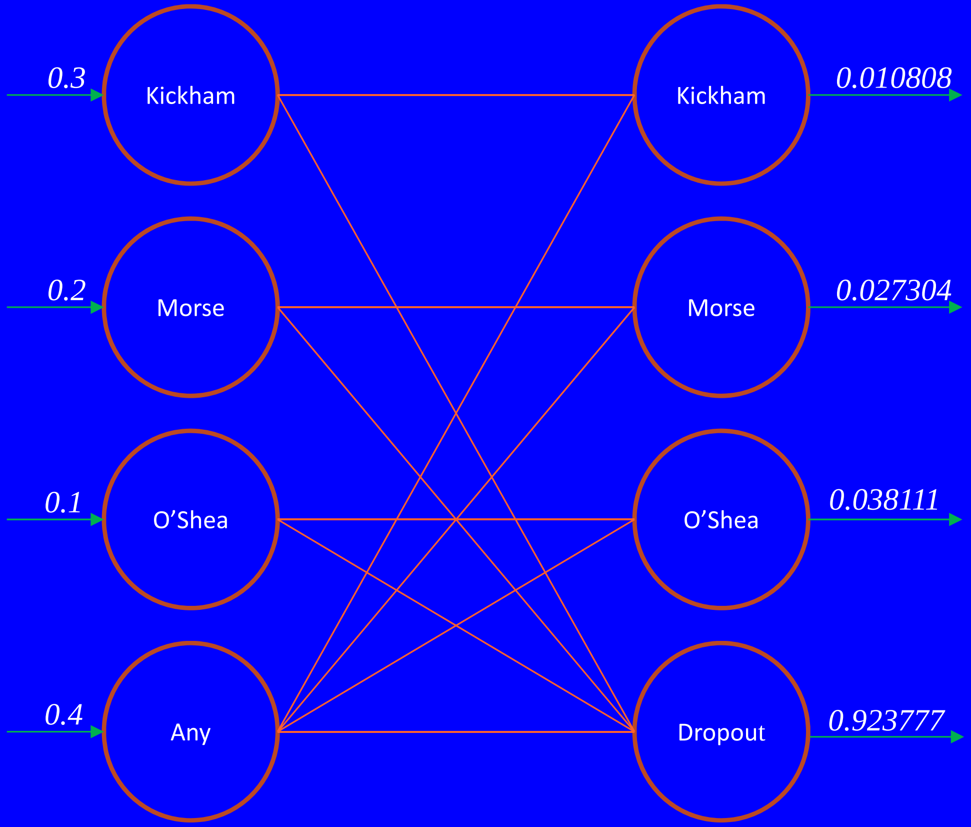


Optimization: Public Housing



Example data from Kaplan EH, A public housing queue with reneging and task-specific servers, *Decision Sciences* 19, 1988.

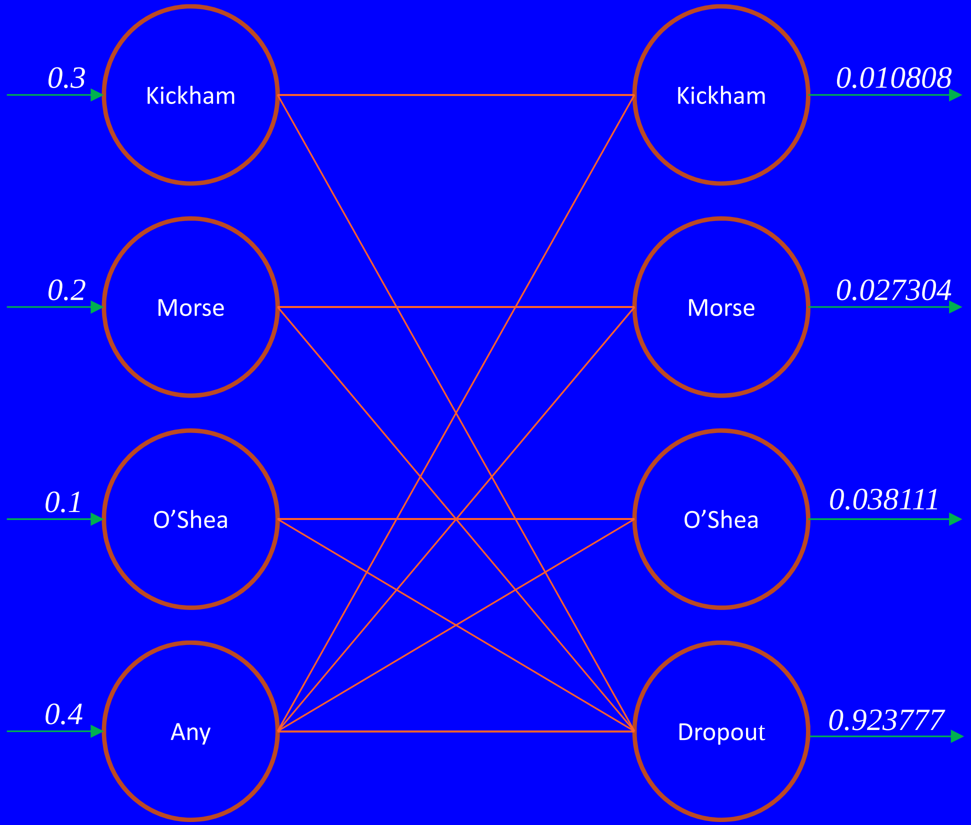
Optimization: Public Housing



Ohm Error ($k=0$) Modified Ohm Error ($k=1$)

$$\begin{array}{c|cccc}
 & Kickham & Morse & O'Shea & Anywhere \\
Kickham & \begin{array}{r}0.004917\\-0.000046\\0.000003\end{array} & 0 & 0 & \begin{array}{r}0.295083\\0.000046\\-0.000003\end{array} \\
Morse & 0 & \begin{array}{r}0.009582\\-0.000134\\0.000009\end{array} & 0 & \begin{array}{r}0.190418\\0.000134\\-0.000009\end{array} \\
O'Shea & 0 & 0 & \begin{array}{r}0.007883\\-0.000072\\0.000004\end{array} & \begin{array}{r}0.092117\\0.000072\\-0.000004\end{array} \\
Anywhere & \begin{array}{r}0.005891\\0.000046\\-0.000003\end{array} & \begin{array}{r}0.017722\\0.000134\\-0.000009\end{array} & \begin{array}{r}0.030228\\0.000072\\-0.000004\end{array} & \begin{array}{r}0.346159\\-0.000252\\0.000017\end{array}
\end{array}$$

Optimization: Public Housing



Ohm Error ($k=0$)

Modified Ohm Error ($k=1$)

$$\begin{array}{c|cccc|}
 & \mathit{Kickham} & \mathit{Morse} & \mathit{O'Shea} & \mathit{Anywhere} \\
\mathit{Kickham} & \begin{matrix}0.004917\\ -0.000046\\ 0.000003\end{matrix} & 0 & 0 & \begin{matrix}0.295083\\ 0.000046\\ -0.000003\end{matrix} \\
\mathit{Morse} & 0 & \begin{matrix}0.009582\\ -0.000134\\ 0.000009\end{matrix} & 0 & \begin{matrix}0.190418\\ 0.000134\\ -0.000009\end{matrix} \\
\mathit{O'Shea} & 0 & 0 & \begin{matrix}0.007883\\ -0.000072\\ 0.000004\end{matrix} & \begin{matrix}0.092117\\ 0.000072\\ -0.000004\end{matrix} \\
\mathit{Anywhere} & \begin{matrix}0.005891\\ 0.000046\\ -0.000003\end{matrix} & \begin{matrix}0.017722\\ 0.000134\\ -0.000009\end{matrix} & \begin{matrix}0.030228\\ 0.000072\\ -0.000004\end{matrix} & \begin{matrix}0.346159\\ -0.000252\\ 0.000017\end{matrix}
\end{array}$$

Pr{Dropout}

Kickham:	0.983611
Morse:	0.952089
O'Shea:	0.921174
Anywhere:	0.865397

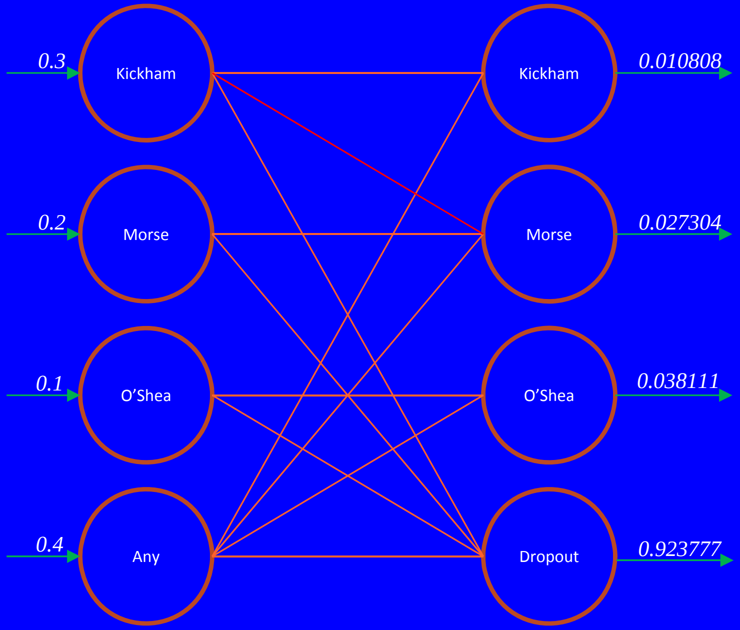
$\Rightarrow$ $\mathit{Max} - \mathit{Min}$

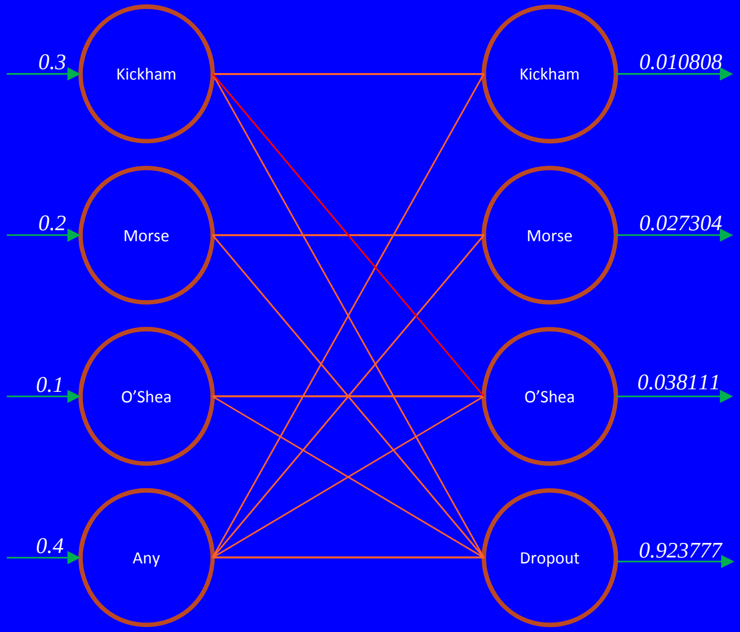
0.118214

Optimization: Public Housing

- 12% difference in dropout rates across applicant types
- Suppose could add just one more link (that is, allow applicants stating preference for only one specific project to expand their choice set to consideration of a second project)
- Which single new link would have greatest impact on difference in dropout rates across applicant types?

Optimization: Public Housing





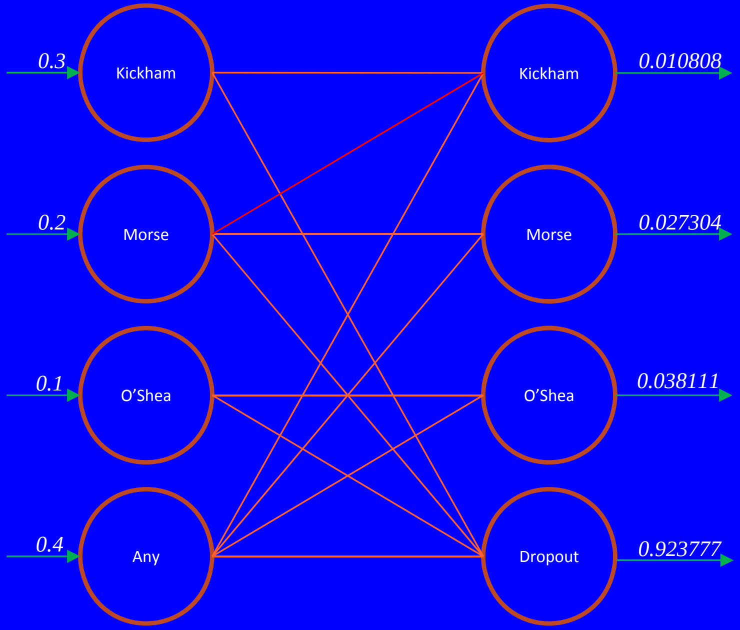
$Pr\{Dropout\}$

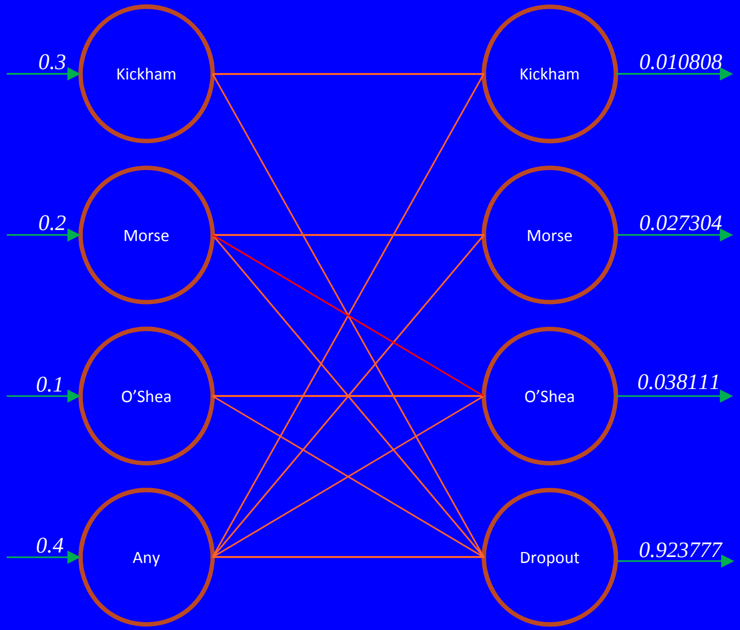
$Kickham$:	0.952767		$Max - Min$
$Morse$:	0.968314	$\Rightarrow$	
$O'Shea$:	0.922266		**0.08817**
$Anywhere$:	0.880144		

$Pr\{Dropout\}$

$Kickham$:	0.935822		$Max - Min$
$Morse$:	0.953278	$\Rightarrow$	
$O'Shea$:	0.950807		**0.060043**
$Anywhere$:	0.893235		

Optimization: Public Housing





$Pr\{Dropout\}$

$Kickham$:	0.987215
$Morse$:	0.940234
$O'Shea$:	0.921389
$Anywhere$:	0.868567

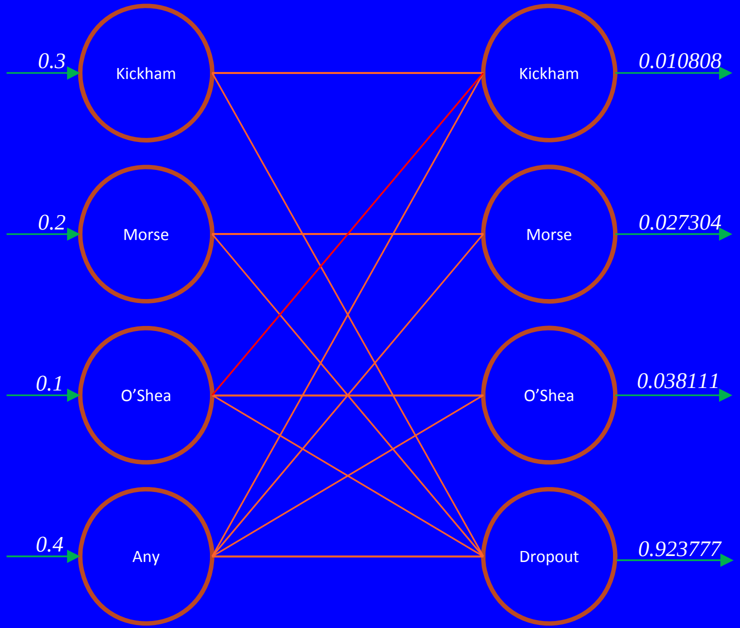
$\Rightarrow$ $Max - Min$ $\mathbf{0.118648}$

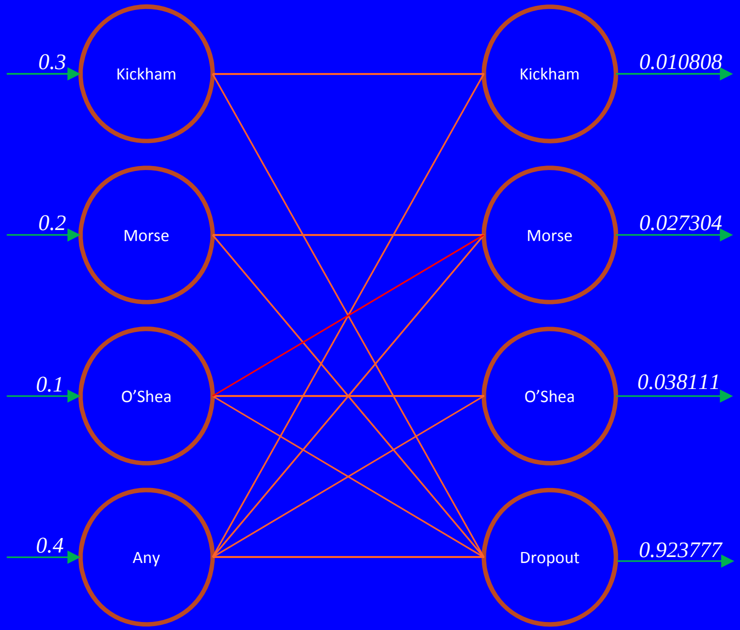
$Pr\{Dropout\}$

$Kickham$:	0.983872
$Morse$:	0.899788
$O'Shea$:	0.942994
$Anywhere$:	0.885896

$\Rightarrow$ $Max - Min$ $\mathbf{0.097976}$

Optimization: Public Housing





$Pr\{Dropout\}$

$Kickham$:	0.985545
$Morse$:	0.952151
$O'Shea$:	0.908439
$Anywhere$:	0.867099

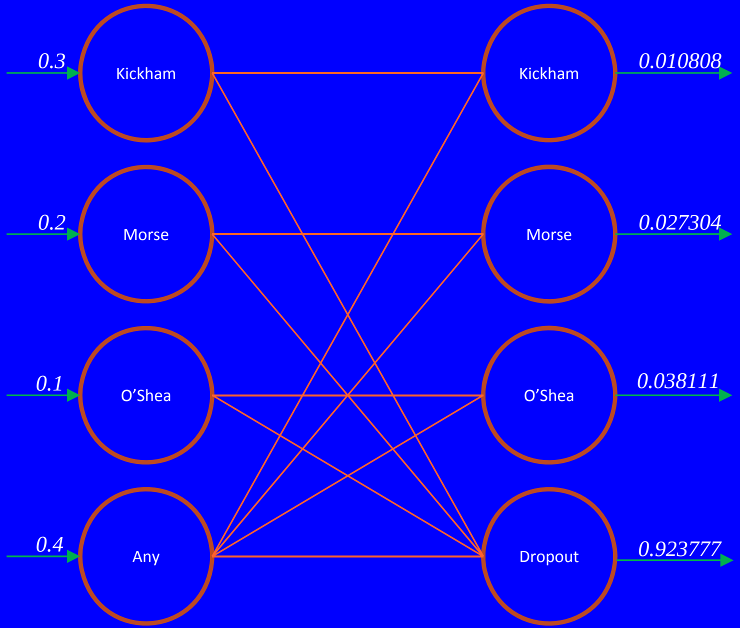
$\Rightarrow$ $Max - Min$ **0.118446**

$Pr\{Dropout\}$

$Kickham$:	0.983680
$Morse$:	0.958492
$O'Shea$:	0.884879
$Anywhere$:	0.871217

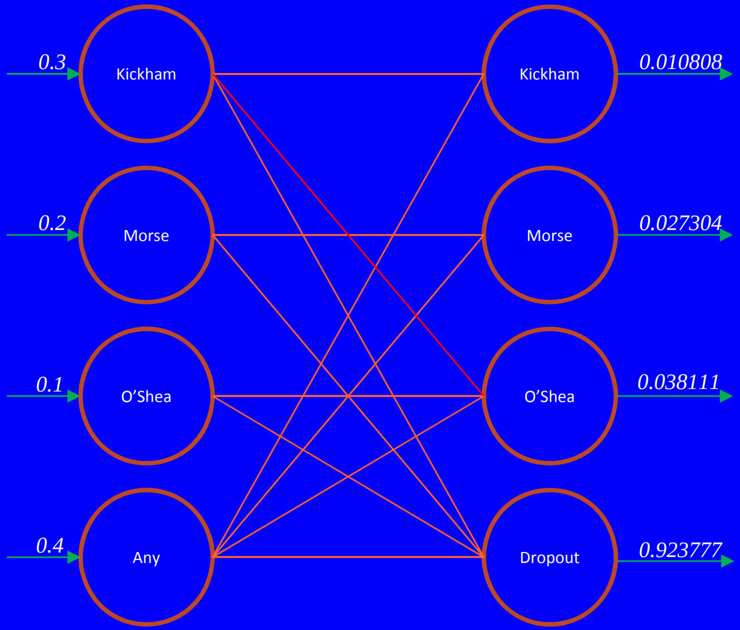
$\Rightarrow$ $Max - Min$ **0.112463**

Optimization: Public Housing



$$Max(\Pr\{Dropout\}) - Min(\Pr\{Dropout\})$$

$$\mathbf{0.118214}$$



$$Max(\Pr\{Dropout\}) - Min(\Pr\{Dropout\})$$

$$\mathbf{0.060043}$$

Summary

- We started with a problem dating back (at least) to 1984
- We inherited a beautiful solution from Adan and Weiss (2012) that is computationally very difficult for all but very small problems
- We imagined the matching process between customers and servers as electric current flowing through a circuit, modeled this using Ohm's Law, and got exact answer for almost complete graph
- We found a pattern in the Adan/Weiss formula for $n - 2$ graphs that could be represented using a shadow graph that in turn directed correction terms to Ohm's Law; this procedure is exact conditional on knowing a constant k, but leads to easy bounds/approximations
- Showed via simulation that approach works well for more general graphs
- Showed a sample optimization problem one could pursue with this method