Dynamic Matching in Overloaded Systems

Jacob D. Leshno*

Abstract

In many assignment problems items arrive stochastically over time. When items are scarce, agents form an overloaded waiting list and items are dynamically allocated as they arrive; two examples are public housing and organs for transplant. Even when all the scarce items are allocated, there is the efficiency question of how to assign the right items to the right agents. I develop a model in which impatient agents with heterogeneous preferences wait to be assigned scarce heterogeneous items that arrive stochastically over time. Social welfare is maximized when agents are appropriately matched to items, but an individual impatient agent may misreport her preferences to receive an earlier mismatched item. To incentivize an agent to avoid mismatch, the policy needs to provide the agent with a (stochastic) guarantee of future assignment, which I model as putting the agents in a priority buffer-queue. I first consider a standard queue-based allocation policy and derive its welfare properties. To determine the optimal policy, I formulate the dynamic assignment problem as a dynamic mechanism design problem without transfers. The resulting optimal incentive compatible policy uses a buffer-queue of a new queueing policy, the uniform wait queue, to minimize the probability of mismatching agents. Finally, I derive a policy which uses a simple rule: giving equal priority to every agent who declines a mismatched item (a SIRO buffer-queue). This policy is optimal in a class of robust mechanisms and has several good properties that make it a compelling market design policy recommendation.

*Microsoft Research Labs New England, jaleshno@microsoft.com

1 Introduction

From nursery schools to nursing homes, waiting lists are a common tool for allocating scarce goods that are not allocated (entirely) by price[1]. If the demand for the good exceeds its supply so much that some people waiting will never be assigned, I will say that the system is overloaded. For example, the Chicago public housing authority presides over approximately 20,000 apartments. When one of these apartments becomes vacant it is assigned to the next applicant on the waiting list of 60,000 applicants who the housing authority allows to register at any given time.[2]

The Chicago public housing authority thus faces a dynamic allocation problem in an overloaded system. In a dynamic matching problem, agents with heterogeneous preferences are to be allocated heterogeneous items that arrive stochastically over time, without the use of monetary transfers. The stochastic arrival adds two elements to the standard discrete allocation problem. First, unlike common assignment problems, items are allocated dynamically as they arrive. Second, an agent's allocation needs to specify not only the item she receives, but also the time at which she receives it. In this paper, I introduce the dynamic allocation problem described above and solve it by presenting new and efficient dynamic allocation mechanisms.

To make proper use of its limited housing stock, the public housing authority should not only make sure that all apartments are assigned, but also try to match apartment locations to the preferences of applicants. The stock of public housing apartments is spread throughout the city, some in the northern part of the city and some in the south.

[1]For example, many teams in the National Football League ration their limited supply of season tickets by a waiting list, instead of increasing the price to clear the market.

[2]The Chicago public authority currently does not accept new applicants to its waiting list. From time to time the authority admits new applicants to its waiting list through a lottery. Once registered, the eligibility of the applicant is verified and a priority class is determined (for disaster victims or domestic violence victims), as well as the appropriate number of rooms (based on the family size). Some applicants (but not all) are allowed to indicate a geographic preference area. Applicants are offered units based on a priority system. They are removed from the waiting list if they refuse a unit without good cause, or if they refuse a second unit (with or without cause).

Applicants have different location preferences, so assigning each apartment as it becomes available to the next person in line may be inefficient, since it can assign an applicant who prefers a northern apartment to a southern one, and vice versa.

To efficiently allocate apartments the public housing agency needs to account for the applicants' preferences, which the public housing authority cannot verify. Preferences of applicants can arise for diverse reasons, such as work location or proximity to family and friends, and so they may not be observable to the housing authority if it does not elicit preference information or allow choice. But apartments arrive over time[3] and applicants are assigned both an apartment at a certain location as well as an interval of waiting time. It is socially efficient for an applicant to decline an apartment that does not suit her very well, allowing it to be assigned to an applicant who prefers it, as one of the two will have to keep waiting in any event. However, if the waiting time is long, an applicant who considers only her own waiting cost may prefer to take an earlier mismatched apartment rather than wait for an apartment she prefers.

To illustrate the model, let us suppose apartments differ only by location, north or south. An apartment and an applicant are said to be mismatched if the applicant would prefer the other location. Apartments become available stochastically over time, and so the allocation needs to be determined sequentially. Suppose that a northern apartment arrives. The mechanism approaches applicants sequentially, trying to find a northern applicant to whom the current northern apartment should be assigned. Not knowing the applicants' preferences, the mechanism may first offer the apartment to a southern applicant (e.g. offer the apartment first to the applicant next on the waiting list, whose locational preferences are not known, and who may accept or reject it). To avoid mismatching a southern applicant (who could choose to accept the northern apartment), the mechanism needs to offer a southern applicant an acceptable alternative: if the applicant

[3]Public housing apartments are made available to new applicants only when current tenants decide to vacate their apartment, for details see Kaplan (1984, 1986).

declines the current (northern) apartment, she gets a claim over apartments that will arrive in the future. For example, an applicant who declines the current northern apartment might be promised the next southern apartment to arrive. The southern applicant thus faces a choice between accepting the current northern apartment or declining it in order to get a southern apartment after some extra wait, and will prefer to decline the current northern apartment if the waiting time is below some threshold.

After the mismatched applicant declines the item, the mechanism is free to continue searching for an applicant who prefers the apartment. The mechanism may encounter numerous mismatched applicants before finding an applicant who prefers the apartment, and mismatched applicants accumulate in a "buffer-queue", each holding a guarantee for future assignment. The mechanism is constrained by past guarantees made to applicants in the buffer-queue. For example, if the next southern apartment to arrive is already committed to an applicant in the buffer-queue, the mechanism can at best promise to give the next applicant who chooses to enter the buffer-queue the second southern apartment to arrive. As the buffer-queue grows large the mechanism is bound by more commitments to applicants. When the buffer-queue is "full" (i.e. the mechanism cannot make an offer that would incentivize a mismatched applicant to decline an apartment), the mechanism is forced to assign the apartment to the next applicant, resulting in a potential misallocation.

In this paper, I set up a model to study the general problem of matching items to agents in an overloaded waiting list. I show that the dynamics of the assignment mechanisms can be captured by a relatively simple Markov chain. The states of the Markov chain correspond to possible states of the buffer-queue, each state describing the information the mechanism holds on agents who reported their type but have not yet been assigned. Transition between states occurs when an item arrives or an agent is asked to report her type.

The agent's incentive constraint to report truthfully is given by this Markov chain. The

paper considers mechanisms where the agent's report determines whether she immediately gets the current item or whether she will take a position in the buffer-queue and receive her preferred item after some additional wait. Her position in the buffer-queue is a guarantee of future assignment computed by the random evolution of the system, as specified by the Markov chain. An agent will report to be mismatched if she prefers this stochastic assignment over the immediate mismatched item. The assignment policy of the buffer-queue determines the "size" of the buffer-queue, or how many agents are willing to join before the buffer-queue is full.

The social planner's problem can be greatly simplified when the system is overloaded and agents have identical waiting costs: welfare is maximized by minimizing the probability of misallocating an item. In each period, one agent is assigned an item and the rest have to wait. Since the system can only shift waiting costs between agents,[4] the sum of waiting costs is constant across assignments. Assuming that agents join the waiting list exogenously when they become eligible, I am able to abstract from the agents' arrival process. The Markov chain describing the buffer-queue is a full description of the policy and allows me to calculate welfare. Welfare is determined by the probability that the system misallocates an item, which in turn can be calculated by the probability that the buffer-queue is full and the system is forced to give an item to the next agent at random.

To illustrate this analysis, I first use the model to calculate the welfare of a common policy with separate waiting lists for two items, in which agents are allowed to decline mismatched items. I show that this can be modeled as a buffer-queue policy where the assignment policy is "First In First Out" (FIFO). Using the Ergodic distribution of the Markov chain, I calculate the misallocation probability and determine the resulting welfare. I also show that welfare is fully determined by the size of the buffer-queue. The policy that maximizes welfare will therefore be given by a queueing policy that generates

[4]Throughout the paper I assume that the mechanism must assign each item in the period it arrives.

a buffer-queue with the maximal number of acceptable positions.

I continue to generalize the analysis and derive the optimal buffer-queue policy. By the previous steps, the problem of finding the welfare maximizing policy can be reduced to finding the policy for the buffer-queue that maximizes its incentive compatible size. A position in the buffer-queue is acceptable if the expected wait is below some critical wait threshold. While the expected wait at a given position depends on the policy in a complicated manner, I use Little's Law to show that the expected wait at a random position depends only on the probability that the buffer-queue is full. This implies that decreasing the expected wait from one position would necessarily increase it at another position. Thus, to make as many positions as possible acceptable, the expected wait should be spread equally between the different positions. I show how this can be done in the two-item case by introducing a new queueing policy, the Uniform Wait (UW) queueing policy. An agent who joins the uniform wait queue faces the same expected waiting time, regardless of the number of agents already on the queue. Under the UW policy, the buffer-queue can hold almost twice as many agents as the FIFO buffer-queue, eliminating about a third of the misallocation.

While the UW policy is optimal, it relies heavily on the exact specification of the environment. The mechanism designer needs to know the precise parameters of the environment, and a mechanism set for the wrong parameters performs poorly. In addition, the UW mechanism incentivizes agents to report truthfully only when the agents have correct beliefs: while the agent joining first gets the item after a below average wait if no other agents join, she faces a long wait if other agents join behind her. The first agent thus faces a gamble, which she accepts if she has correct beliefs but will prefer to mismatch under pessimistic beliefs. This also implies that the mechanism is not ex-post incentive compatible, as when the queue becomes long the agent in the first position would prefer an immediate mismatch to a position in the buffer-queue.

I therefore introduce the "Service In Random Order" (SIRO) buffer-queue policy, and show that it is an approximately optimal, simple and parameter-free mechanism. Under the SIRO policy, each agent in the buffer-queue has an equal probability of receiving an arriving item, regardless of her position in the queue. I show that the SIRO policy is the optimal solution when the designer is required to choose a robust mechanism, and that this policy has a number of other appealing properties.

The SIRO policy strictly improves upon the FIFO policy by giving to the marginal agent greater incentive to join the queue, as being in the last position in a SIRO queue denotes a higher probability of receiving an assignment than does being the last position in the FIFO queue. Moreover, SIRO is ex-post incentive compatible when agents are homogenous. SIRO is simple to implement and agents will converge to truth-telling when following simple learning dynamics. Finally, I advocate for the SIRO policy, showing that it achieves close to the optimal welfare under the specification of the model, and outperforms both the FIFO and the UW policies in more general environments.

Many real-life assignment problems have a dynamic component, and the use of waiting lists is common, but the literature on dynamic allocation problems without monetary transfers is limited. The market design literature mainly focuses on static allocation problems, such as school choice, where all items and agents are allocated at once. As far as I know this paper is the first to model the dynamic matching problem in waiting lists.[5] The recent literature on dynamic mechanism design considers dynamic allocation problems where monetary transfers are allowed,[6] whereas this paper focuses on the matching between agents and items when monetary transfers are not allowed. The queueing literature studies the allocation of waiting time to agents, focusing on the case where all items

[5] Ünver (2010) investigates the dynamic kidney exchange problem under full information, where patient-donor pairs, whose type is known to the mechanism, arrive over time. See also Abdulkadiroglu and Loerscher (2007); Kurino (2009); Kennes, Monte, and Tumennasan (2011) for other dynamic considerations in market design.

[6] For a review of the dynamic mechanism design literature see Bergemann and Said (2010).

are homogenous.[7] The focus of this paper is matching between agents and items when agents have heterogeneous valuations over items and can misreport their preferences.[8][9] It is also distinct from discrete allocation problems in that the allocation occurs dynamically over time.[10]

The remainder of the paper is organized as follows. Section 2 sets up the model. Section 3 proceeds to illustrate the model by analyzing a common waiting list policy—holding parallel waiting lists and allowing agents to decline items. This analysis is generalized in section 4, where I present the optimal mechanism—the Uniform Wait (UW) buffer-queue mechanism. Section 5 presents the Service In Random Order (SIRO) buffer-queue mechanism and shows that it is an optimal parameter-free mechanism.

2 A model for overloaded waiting lists

In this section we set up a simple model in which agents with heterogeneous preferences join an overloaded waiting list to receive heterogeneous items. We show that if all agents have equal costs of waiting, welfare is determined purely by the matching between agents and items. This enables us to abstract away from the arrival process of agents. In the following sections we show how this model allows us to achieve a tractable analysis of

[7] The queueing literature considers the case where agents strategically choose between servers to minimize their waiting time (see Bell and Stidham Jr (1983)), but assumes that agents' preferences are fully captured by their waiting time. This paper adds to this literature by allowing agents to have heterogeneous valuations for service at different servers. In addition, this paper not only considers a fixed queueing policy, but also derives the optimal policy.

[8] Caldentey, Kaplan, and Weiss (2009) investigate the matching rates between multiple servers and multiple customer classes when each server is compatible with only a subset of the customer classes, but they do not allow customers to choose servers. In this paper agents choose with which server they are compatible, and the agents' choice is allowed to depend on the expected wait for each server. Also see Kaplan (1988); Adan and Weiss (2010); Talreja and Whitt (2008)

[9] Su and Zenios (2006); Zenios (1999) model the waiting list for kidney transplants and investigate how to prevent agents from declining low quality kidneys. This paper considers horizontal preferences where it is socially efficient for agents to decline low value items, as these items are of high value to other agents.

[10] Barzel (1974) considers the allocation through queues as a static allocation problem in which agents are allocated items as well as a non-stochastic waiting time. In this paper I show that stochastic fluctuations in waiting time cause misallocation of items.

assignment mechanisms.

Agents exogenously[11] arrive and join the waiting list, each waiting to be allocated a single item. The waiting list is overloaded[12], that is there at least M agents on the waiting list at any time. Agents have private preferences over items given by their private type α or β. We refer to the non-preferred other item as the *mismatched* item. Agents receive a value of 1 from their preferred item and a lower value $v < 1$ from their mismatched item. All agents are risk neutral and pay a waiting cost c per period until they are assigned an item and removed from the system. The initial probability that an agent on the waiting list is of type α is p_α (otherwise, of type β with probability $p_\beta = 1 - p_\alpha$). We denote the set of agents by $\mathcal{A}$.

Every period $t \geq 0$ a single item $x_t \in \{A, B\}$ arrives. The probability that the item x_t is of kind A is p_A (otherwise it is of kind B with probability $p_B = 1 - p_A$).[13] Each item must be assigned in the period in which it arrives:

Definition 1. An allocation is an injection $\mu : \{t \geq 0\} \to \mathcal{A}$ allocating each item x_t to a distinct agent $a = \mu(t)$.

The designer's aim is to implement an allocation that would maximizes the total utility gains generated by the limited supply of items. While an individual agent's utility depends both on the waiting costs and the value of item assigned, the goal of the designer is only to maximizes the assignment value by matching agents to their preferred items:

[11] While this assumption is unsuitable for a queue, it is a reasonable approximation for a waiting list. In a typical queue agents signal their value for items by costly waiting in line (Barzel (1974)). However, in a waiting list agents do not bear any additional cost from being on the waiting list (Lindsay and Feigenbaum (1984); Cullis and Jones (1986)). In many of the applications of interest signing up to the list is almost costless, and agents join the waiting list despite having a low probability of getting any assignment (Kaplan (1988); Su and Zenios (2005)). I therefore abstract away from incentives to join the waiting list and assume that agents will join the waiting list when they exogenously become eligible to join.

[12] Long waiting lists are common in practice. For example, the Chicago public housing authority currently does not accept new applicants, as the 60,000 person waiting list is full. The waiting list for kidney transplants is growing from year to year, and currently holds more than 80,000 patients.

[13] I assume that arrival rates are i.i.d. over time for simplicity. In the context of public housing Kaplan (1986) justifies this assumption by observing that the move-out (of current tenants) process is a Poisson process.

Claim 2. For any finite time horizon and under any overloaded arrival process the difference in sum of agents' utilities between allocations depends only on the matching of item kinds and agent types.

This and other omitted proofs can be found in the appendix.

We say that *misallocation* happened in period t if agent $a = \mu(t)$ was assigned x_t which is a's mismatched item. Given the previous result the aim of the social planer is maximize welfare by minimizing misallocation. Since every misallocated item generates a utility of v instead of 1 the long run welfare loss from misallocation is defined as follows:

Definition 3. The long run welfare loss from misallocation under an allocation μ is

$$(1 - v)\xi,$$

where $\xi = \liminf_{T\to\infty} \frac{1}{T} \sum_{t=t_0}^{T} \xi_t$ is the long run average probability of misallocation.

We will focus on the case $p_A = p_\alpha$, that is there is an equal proportion of agents as of their preferred items. In this case there is no structural mismatch and the welfare loss from misallocation results only from the preference revelation problem. When there are at least $M \gg 0$ agents on the waiting list at any given time a mechanism with full information would assign almost every item to an agent who prefers it, as shown by the following result (proven in section 3.3):

Proposition 4. *Suppose that $p_A = p_\alpha$ and that the waiting list holds at least M agents at any given period. Then a mechanism with full information can achieve an expected misallocation probability of $\xi^{FB} \leq \frac{1}{2M}$.*

The mechanism designer's goal is therefore to find a matching agent for each arriving item. When an item x_t arrives the mechanism knows the history: the items that arrived in previous periods, the agents who were assigned items, agents who reported their type to

the mechanism (i.e. made a choice) and agents who joined the system but did not report their type (i.e. they did not make any choice yet). Information about past assignments is irrelevant, as agents who were assigned items are gone from the system. All agents who did not make a choice are interchangeable[14]. We assume that the mechanism encodes all of the relevant information using a state space S, and each history is mapped to a state $s \in S$.

Given the information state $s \in S$ the mechanism can take two possible actions. The mechanism can either allocate the current item to an agent or solicit more information by asking an agent to report her type. When the mechanism assigns the item x_t to agent $a \in \mathcal{A}$ the period ends. We set $a = \mu(x_t)$, remove a from the system, and start the next period by drawing the item x_{t+1}. If the mechanism decides to solicits more information another agent will be asked to report her type. Agents choose which report to make, and can either reveal their true type or misreport to induce a more preferred allocation. Given the report of the agent the state is updated to s' and the period continues. Given the new state s' the mechanism again decides between soliciting more information and making an assignment.

Within a period the mechanism may sequentially ask multiple agents to report their type, and sequentially updating the state. The state s is randomly updated according to the reported types, until reaching a state $s \in S$ in which the mechanism decides to assign x_t and the period ends. We require that with probability 1 every period ends after asking a finite number of agents.

We capture the state of the world by $\omega_t \in \{A, B\}^{\mathbb{N}} \times \{\alpha, \beta\}^{\mathbb{N}}$, giving the sequence of future item kinds and the sequence of agent types.[15] We assume that all agents report

[14]The past waiting time of each agents does not provide any relevant information. Since past waiting costs are sunk and do not affect the current value of allocation. Therefore the period in which an agent arrived does not provide any information about the current valuation of the agent. The United Network for Organ Sharing (UNOS) allocates liver transplants to patients on the current severity of their disease, ignoring the waiting time of patients.

[15]We use the fact that all unapproached agents are symmetric and encode the types of agents in the

their type truthfully, and describe the mechanism by a deterministic state machine whose transitions depend on ω_t.

Definition 5. A dynamic mechanism is defined by a state space S and a transition function

$$\mathcal{T} : s_t \times \omega_t \mapsto a \times s_{t+1} \times \omega_{t+1}$$

The transition function $\mathcal{T}$ describes that when the mechanism starts period t at a history corresponding to state s_t and the state of the world is ω_t the item x_t will be assigned to agent $a = \mu(t)$ and next period will start with state $s_{t+1} \in S$. The mechanism may solicit the type of finitely many agents within period t, and ω_{t+1} encodes the information in the state of the world that the mechanism did not learn yet. Given an initial state s and an initial state of the world ω_0 the transition function produces a complete assignment μ. The transition function $\mathcal{T}$ therefore defines a deterministic mechanism $\mathcal{M} : \omega_0 \times s \mapsto \mu$.

Note that if we do not impose any incentive constraints a mechanism that perfectly matches all items is straightforward: keep asking a new agent until a matching agent is found to receive the item. Proposition 8 shows that a similar policy can be implemented when the waiting list is finite, and that the rate of misallocation converges to 0 as the waiting list becomes more overloaded.

A simple example of a mechanism is the sequential assignment which does not allow agents to express their preferences, assigning the k-th item to the k-th agent on the waiting list. This policy can be described by a trivial Markov chain with a trivial state space $S = \{\phi\}$, as regardless of history the item is always assigned to the agent at the head of the waiting list. At any period t a random item is assigned to an agent with a randomly drawn type, so the misallocation probability in every period as well as in the

order in which they are asked. That is $(\alpha, \beta, \beta, ...)$ denotes that the first agent asked will be an α and the second and third agents asked will be β.

long run is:

$$\begin{aligned}\xi^{Rand} &= p_A p_\beta + p_B p_\alpha \\ &= 2p(1-p)\end{aligned}$$

and the welfare loss from this policy will amount to $(1-v)2p(1-p)$.

While this policy is clearly suboptimal, it is often used in practice. In many public housing authorities applicants cannot choose their apartments, and an agent who declines an apartment is penalized by removal from the waiting list,[16] resulting in a random assignment of applicants within a category. While the public housing authority separates candidates into different categories and matches the categories to apartments, the allocation does not consider the agents' private information, for instance location preferences. For example, the housing authority matches the family size to the number of rooms of the apartment, but it does not assign families to their preferred neighborhood or allow them to be located next to friends and family. The welfare loss from this random assignment can be substantial. For example, if there is equal supply and demand of apartments in two geographic locations in the city and agents value a mismatched apartment at 70% of the value of their preferred apartment then this policy will lead to a loss 15% of welfare.

In the following sections we consider the class of policies that offer agents a choice between immediate assignment or waiting in a buffer-queue for the preferred item. We continue our analysis by considering a common waiting list policy that allows agents to express their preferences. The analysis will introduce the principles that we will subsequently use to derive the policy that minimizes misallocation and maximizes welfare.

[16]The housing authority allows agents to decline housing only with "cause", for example a physical disability.

3 Benchmark policy – FIFO

In this section we show how the model can be used to analyze a common waiting list policy—allowing to decline items and keep their place in line. The analysis in this section serves two functions. First, we show how a common policy can be described within the model and calculate its welfare. Second, the analysis allows us to present the tools that will later be generalized and used to derive the optimal policy.

We start from a description of the policy as an indirect mechanism. There are two infinite parallel waiting lists, one for A and one for B, both listing in order all agents that will ever join the waiting list in order of arrival. Initially every agent on the waiting list is of type α with probability p_α and of type β with probability $p_\beta = 1 - p_\alpha$. Every period one item of a random kind arrives, with prob p_A an A item arrives and with prob $p_B = 1 - p_A$ a B item arrives. Items are offered to agents on the waiting list according to a First In First Out ($FIFO$) order. When an agent is offered an item she can either take the item and be removed from the system, or decline the item and keep her place in the waiting list for the other item[17]. Declined items are offered to the next agent on the waiting lists until an agent takes the item. The period ends when the item is taken. All agents pay c for every period they spend on the waiting list, and get a value of 1 from their matching item (A for α types, B for β) or a value of v from their mismatched item (B for α, A for β). That is, the payoff an agents who waited w periods and got his matching item is $1 - c \times w$. Agents know their position in the waiting list when they decide whether to take or decline an item.

In this section we will show how this mechanism can be described as a dynamic direct mechanism and follow up on the previous analysis to derive its welfare properties.

[17]When preferences are stationary, an agent who declined an item will decline it again in the future, and therefore it does not matter whether an agent that declined a B is removed from the B waiting list.

3.1 The agent's decision

Assuming that there is no cost to sign up to a waiting list, agents should always sign up for both waiting lists. The first time an agent faces a choice when she is offered an item. When an agent is offered her preferred item she will always take it. When offered a mismatched item the agent needs to choose between taking the mismatched item immediately, or declining it and incurring an extra wait to get a preferred item. The agent will decline a mismatched item only if the expected extra wait for a preferred item is short enough.

Consider an α agent who is offered a B when she is k-th in line for an A. Taking the mismatched B item immediately will yield utility of

$$U_\alpha(current\ B) = v$$

and the value of being the k-th in line for an A is

$$\mathbb{E}\left[U_\alpha(wait\ for\ A \,|\, k\text{-th}\, in\ A\ line)\right] = 1 - c \cdot \frac{k}{p_A}$$

as the agent will get an A item that will give her a value of 1, but she will have to wait till the k-th A item will arrive, which takes in expectation $k \times \frac{1}{p_A}$ periods.

Therefore, an α agent will be willing to decline a B when her position is k if and only if[18]

$$1 - c \cdot \frac{k}{p_A} \geq v$$

which is true when

$$k \leq K_\alpha \equiv \lfloor p_A \frac{1-v}{c} \rfloor$$

Note that $\frac{1-v}{c}$ is the expected waiting time that will make the agent indifferent between

[18] We assume that agents decline the mismatched item when they are indifferent.

her matching item and an immediate mismatch, and p_A transforms waiting time units from periods to A-arrivals.

As more α agents who decline B items accumulate to wait for an A item, the longer extra wait another declining agent will have to incur if she declines a B as well. The agent's impatience dictates a maximal number $K_\alpha = \lfloor p_A \frac{1-v}{c} \rfloor$ of α agents who declined a B and are still waiting for an A, and likewise a maximal number $K_\beta = \lfloor p_B \frac{1-v}{c} \rfloor$ of β agents who declined an A and are still waiting for a B.

3.2 The Markovian model

To calculate the welfare of this policy we need to know the probability that a B item will be allocated to an α agent. By the previous section, this depends on the probability of accumulating K_α agents that declined a B. In this section we give a Markovian model that will capture the dynamics of the system and will allow us to calculate the misallocation probability.

A few observations allow us to get a simple representation of the system dynamics. First, when the policy allows to decline items any agent that becomes eligible to join the waiting list will join both the A and B waiting lists. Therefore the same agents will be on the two lists in the same order, except for agents who declined an item. At any point in time we can split the agents that are still unmatched into two categories: agents who weren't ever offered an item, which we will refer to as **new agents,** and agents who previously declined an item, which we will refer to as **waiting agents**.

We can describe the system by two sets of queues. The **offer queue** holds all the new agents in the order of their arrival, representing the identical part of both waiting lists. The two **assignment queues**, one for A and one for B, hold waiting agents who previously declined an item. The assignment queues represent the front of the waiting lists, containing agents that are at the front of one of the waiting lists and were removed

from the other waiting list after declining an item. When an A item arrives it will first be offered to the agent at the front of the A assignment queue (who is the agent at the head of the A waiting list). If the A assignment queue is empty, the item will then be offered to the agent at the head of the offer queue. When there are $k \geq 0$ agents on the B assignment queue, the agent at the head of the offer queue is given a choice between taking the current A item, or being placed in the $k+1$ position in the B assignment queue (which is the $k+1$ position in the B waiting list).

An example will be helpful to illustrate the dynamics of the mechanism. Consider the system at the beginning of period t where there are no unassigned agents that declined an item and both assignment queues are empty (i.e. the A and B waiting lists are identical). Let the first agents on the offer queue be of types $\alpha, \alpha, \beta, \alpha, \alpha, \alpha, \ldots$ and assume that $K_\alpha = 3$. Assume that a B item arrives in period t. As the assignment queue is empty, the B item is offered to the first agent in the offer queue (who is the first agent on the B waiting list). This agent is an α type, and since $K_\alpha \geq 1$ she chooses to decline the B and take the first position in the A assignment queue[19]. The B item will then be offered to the next agent in the offer queue, who is another α type. Since $K_\alpha \geq 2$ the agent declines the B item and takes the second position in the A assignment queue. The B item will then be offered to the next agent in the offer queue, who turns out to be a β type. The β agent takes the B item, ending period t.

Assume that in period $t+1$ an A item arrives. This item will be assigned to the first agent in the A assignment queue, ending period t with one α agent left in the A assignment queue.

Assume that in period $t+2$ another B item arrives. As the B assignment queue is empty, the B is offered to the agent at the head of the offer queue. This agent is of type α. As $K_\alpha \geq 2$ this α agent chooses to decline the B item and take the second position

[19]Note that this agent keeps the same place on the A waiting list, the "move" to the assignment queue represents that she is no longer on the B waiting list.

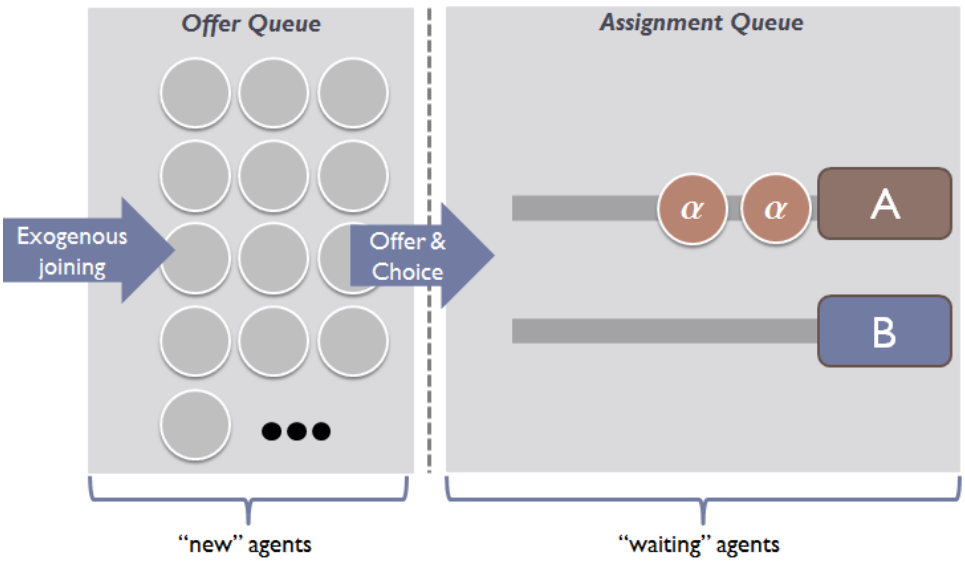


Figure 3.1: Illustration of the waiting list as an offer-queue and an assignment-queue.

in the A assignment queue. The B item is then offered to the following agent, who is an α as well. Since $K_\alpha \geq 3$ this α agent chooses to decline the B item and take the third position in the A assignment queue. Still in period $t+2$, the B item is then offered to the following agent in the offer queue, who is also an α. Since $K_\alpha < 4$ this α agent will take the B item instead of the fourth position in the A assignment queue. Period $t+2$ ends with the last α agent taking the B and 3 α agents waiting on the A assignment queue.

To analyze this mechanism we first represent it as a dynamic revelation mechanism. When an agent declines an item and joins the assignment queue she reveals her type to the mechanism. In the beginning of every period the mechanism knows the identity of the agents in the assignment queue, and knows nothing, beyond the prior distribution, about the agents in the offer queue. The mechanism learns the kind of item that arrived and seeks a matching agent. If the assignment queue for the object holds an agent, who must be a matching type, the system allocates the item to the agent at the head of the assignment queue. If the item's assignment queue is empty the mechanism approaches a new agent a that is at the head of the offer queue and asks her to report her type. If the agent a reports to be of a matching type she is allocated the item and the period ends. If

the agent a reports to be a mismatched type the mechanism checks the number of agents in the assignment queue for the other item. If the length of the assignment queue is less than K (the respective K_α or K_β we calculated), the mechanism puts the agent at the end of the assignment queue and proceeds to approach another new agent from the offer queue. If the length of the assignment queue is K the agent is allocated the item even though this produces a mismatch.

Using this description we have the following result:

Proposition 6. *The assignment probabilities under the $FIFO$ waiting list where agents are allowed to decline items and keep their place in line can be given by a Markovian transition model with the state space*

$$S = \{-K_\beta, \ldots, -1, 0, 1, 2, \ldots, K_\alpha\}$$

where $k \geq 0$ indicates k agents of type α waiting in the A assignment queue and $k \leq 0$ indicate $|k|$ agents of type β waiting in the B assignment queue.

The transition probabilities P are given by:

$$P\left(s = l | s_{t-1} = k\right) = \begin{cases} p_A & k > 0,\, l = k - 1 \\ p_B p_\alpha^{l-k} p_\beta & k \geq 0,\, k \leq l < K_\alpha, l \neq 0 \\ p_B p_\alpha^{l-k}(p_\alpha + p_\beta) & k \geq 0,\, l = K_\alpha \\ p_A p_\alpha + p_B p_\beta & k = l = 0 \\ p_B & k < 0,\, l = k + 1 \\ p_A p_\beta^{l-k} p_\alpha & k \leq 0,\, k \geq l \geq -K_\beta, l \neq 0 \\ p_A p_\beta^{l-k}(p_\beta + p_\alpha) & k \leq 0,\, l = -K_\beta \end{cases}$$

This Markov chain representation allows us to calculate the probability that an item

will be assigned to a mismatched agent. Each transition of this system captures the events from the beginning of period t until the assignment of x_t that ends the period. Since an α agent is assigned a B item only when the assignment queue is full all transitions that assign a B item to an α agent must end in state $s = K_\alpha$. We can calculate the probability of misallocation directly from the ergodic distribution of this chain, but it will be useful to give another Markovian representation that captures the offer process within a period.

Let us consider a larger state space $\hat{S}$ that describes the system at any point when a new agent is offered the item. In addition to the previously defined states we include sets of states S^A and S^B which represent the system in the middle of a period during the process of offering an A or B item to new agents. A state $(k, B) \in S^B$ indicates that there are k agents in the A assignment queue, the B assignment queue is empty and the current B item is about to be offered to the first agent in the offer queue. A state $(-k, A) \in S^A$ indicates that there are k agents in the B assignment queue, the A assignment queue is empty and the current A item is about to be offered to the first agent in the offer queue. The previously defined states are relabeled as $(k, \phi) \in S^\phi \cong S$ that indicates for $k \geq 0$ that there are k agents in the A assignment queue, the B assignment queue is empty and there is no current item to assign (the period ended, and the next item did not arrive yet). For $k \leq 0$ the state $(k, \phi) \in S^\phi$ indicates that there are $|k|$ agents in the B assignment queue, the A assignment queue is empty and there is no current item to assign.

Every period begins and ends in a state in S^ϕ. We move from a state in S^ϕ when an item arrives. Suppose that the state is (k, ϕ) for $0 < k < K_\alpha$. If an A item arrives it is assigned to the first agent in the A assignment queue, the system moves to state $(k-1, \phi)$ and the period ends. If a B item arrives the system moves to state (k, B). The transitions from state (k, B) will depend on the type of the first agent in the offer queue. If the first agent on the offer queue is of type β she will take the current item, the system moves to state (k, ϕ) and the period ends. If the first agent on the offer queue is of type α she will

decline the item and join the A assignment queue, the system moves to state $(k+1, B)$ and the period continues.

The complete state space is given by

$$\begin{aligned}\hat{S} &= S^{\phi} \cup S^{A} \cup S^{B} \\ &= \{(k,\phi) |\ -K_{\beta} \leq k \leq K_{\alpha}\} \cup \{(-k, A) \,|\, 0 \leq k \leq K_{\beta}\} \cup \{(k, B) \,|\, 0 \leq k \leq K_{\alpha}\}\end{aligned}$$

The transitions of the system are given by:

$$P\left(u|(k,\phi)\right) = \begin{cases} p_A & u = (k-1,\phi) \\ p_B & u = (k, B) \end{cases} \qquad k > 0$$

$$P\left(u|(-k,\phi)\right) = \begin{cases} p_A & u = (-k, A) \\ p_B & u = (-(k-1),\phi) \end{cases} \qquad k > 0$$

$$P\left(u|(0,\phi)\right) = \begin{cases} p_A & u = (0, A) \\ p_B & u = (0, B) \end{cases}$$

$$P\left(u|(k,B)\right) = \begin{cases} p_{\alpha} & u = (k+1, B) \\ p_{\beta} & u = (k,\phi) \end{cases} \qquad 0 \leq k < K_{\alpha}$$

$$P\left(u|(K_{\alpha},B)\right) = \begin{cases} p_{\alpha} + p_{\beta} & u = (K_{\alpha},\phi) \end{cases}$$

$$P\left(u|(k,A)\right) = \begin{cases} p_{\alpha} & u = (k,\phi) \\ p_{\beta} & u = (-(k+1, A) \end{cases} \qquad -K_{\beta} < k \leq 0$$

$$P\left(u|(-K_{\beta},A)\right) = \begin{cases} p_{\alpha} + p_{\beta} & u = (-K_{\beta},\phi) \end{cases}$$

The Markov chain on $\hat{S}$ is depicted in figure 3.2. It can give us the distribution over S if we limit attention to visits to S^{ϕ} states. For calculation purposes, the system $\hat{S}$ has the

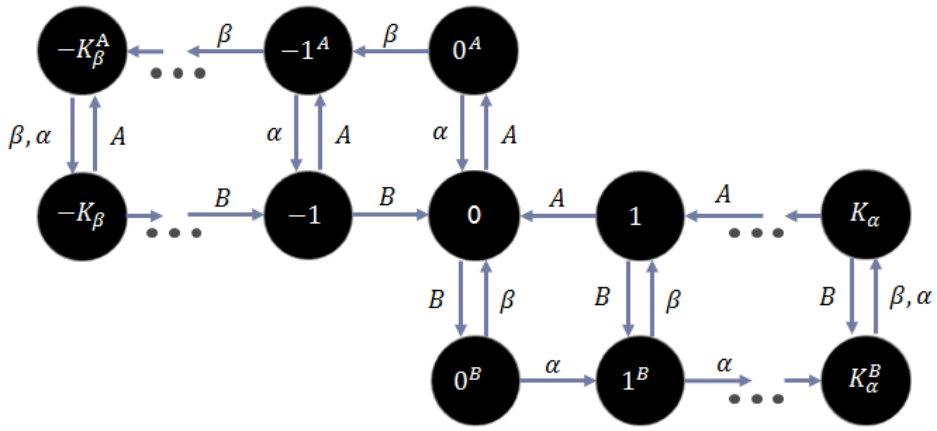


Figure 3.2: The Markov chain for the state space $\hat{S}$

advantage that all moves are between adjacent states. The system $\hat{S}$ also enables us to capture the system at the point at which an agent makes the decision whether to decline or accept an item. Using the state space $\hat{S}$ and flow equations on the Markov chain we derive the stationary distribution which describes the long run behavior of the system:

Lemma 7. *The Markov chain is ergodic and its stationary distribution is*

$$\pi(k) = \begin{cases} \left(\frac{p_\alpha}{p_A}\right)^k p_B \pi(0) & k > 0 \\ \left(\frac{p_\beta}{p_B}\right)^{|k|} p_A \pi(0) & k < 0 \end{cases}$$

and

$$\pi(0) = \begin{cases} \frac{1}{2} \frac{1}{p_B K_\alpha + p_A K_\beta + 1} & p_A = p_\alpha \\ \frac{1}{2} \frac{p_A - p_\alpha}{p_A p_\beta \left(\frac{p_\beta}{p_B}\right)^{K_\beta} - p_B p_\alpha \left(\frac{p_\alpha}{p_A}\right)^{K_\alpha}} & p_A \neq p_\alpha \end{cases}$$

3.3 Misallocation probability

From the stationary distribution calculated in the previous section we can find the misallocation probability. Misallocation happens only when a new agent on the offer queue takes the current item despite it being the mismatched item. When a new agent is offered

an item the system is in a state $s \in S^A \cup S^B$. Denote by $\tilde{\pi}^B(k)$ the stationary probabilities of the state (k, B) conditional on being in one of the states $s \in S^A \cup S^B$, and likewise for $\tilde{\pi}^A(-k)$. An α agent will take a mismatched item only in the state (K_α, B). A β agent will take a mismatched item only in the state $(-K_\beta, A)$. Therefore:

Corollary. *The probability of misallocation under the $FIFO$ policy is*

$$\xi^{FIFO} = p_\alpha \tilde{\pi}^B(K_\alpha) + p_\beta \tilde{\pi}^A(-K_\beta)$$

and if $p_\alpha = p_A = p$ the probability of misallocation simplifies to

$$\xi^{FIFO} = \frac{2p(1-p)}{(1-p)K_\alpha + pK_\beta + 1}$$

This formula has a clear interpretation. Randomly assigning the items would yield a mismatch probability of $2p(1-p)$. In most states the system is able to sort the two types, and allocate the item to the next agent in the offer queue only if that agent is of a matching type. However, when the assignment queue is full the system randomly allocates the item to the next agent, regardless of her type. The greater K_α, K_β are, the longer the assignment queue can be, and the less likely it is that the system will have to randomly allocate the item to the next agent. The denominator in the expression increases as the assignment queues can grow longer, reducing the probability of misallocation.

The previous analysis implies that we should evaluate the policy by its implied probability of misallocation. To isolate the welfare loss from the preference revelation problem we compare the welfare under this policy to the welfare a mechanism with full information can achieve. When the waiting list is infinite the system can allocate each item to a matching agent with probability 1. Using the tools we developed to analyze the dual list policy we can show that this will be approximately true even when the waiting list is finite, but long:

Proposition 8. *Suppose that the waiting list is finite, but includes at least M agents at any point in time and $p_\alpha = p_A = p$. Under full information the system can achieve a stationary mismatch probability*

$$\xi^{FB} \leq \frac{1}{2M}$$

Proof. Under full information the system will have to mismatch an agent only after the entire waiting list was exhausted and no matching agent was found. Consider the policy that requires a mismatched agent to decline an item, unless there already are $M-1$ agents in the assignment queue. As there are at least M agents in the system this policy never exhausts the waiting list. We can analyze this police using the same Markovian model as above by setting $K_\alpha = K_\beta = M - 1$. We find that

$$\begin{aligned} \xi^{FB} &\leq \frac{2p(1-p)}{(1-p)K_\alpha + pK_\beta + 1} \\ &= \frac{2p(1-p)}{(1-p)(M-1) + p(M-1) + 1} \\ &\leq \frac{1}{2}\frac{1}{M} \end{aligned}$$

□

The last proposition means that even when the wait-list is finite, but large, the misallocation will be negligible under full information. In other words, misallocation in the system is fundamentally due to the preference revelation problem. We give the welfare of the $FIFO$ policy in terms of how close to the optimal we get, that is how much welfare loss we get relative to the benchmark of no mismatch.

Proposition 9. *The welfare loss from mismatch is under the $FIFO$ policy is:*

$$\begin{aligned} WFL^{FIFO} &= (1-v)\xi^{FIFO} \\ &= (1-v)\frac{2p(1-p)}{(1-p)K_\alpha + pK_\beta + 1} \end{aligned}$$

where $K_\alpha = \lfloor p\frac{1-v}{c} \rfloor$ *and* $K_\beta = \lfloor (1-p)\frac{1-v}{c} \rfloor$

Corollary 10. *As* $c \to 0$ *the misallocation probability under* $FIFO$ *goes to* 0*, and welfare approaches first best.*

4 Optimal policy

We now turn to consider more general policies and seek the optimal policy. We follow the construction from the previous section and describe mechanisms by their representation as an offer queue and assignment queues. Since all agents in the offer queue are symmetric we can ignore the policy of the offer queue. We consider the class of policies generated by different queueing policies for the assignment queues. Let $(\mathcal{M}, K_\alpha, K_\beta)$ denote a mechanism that places mismatched agents in the assignment queues if they are not full and assigns the item to the next agent in the offer queue otherwise. We refer to K_α as the **size of the** A **assignment queue**, it is the maximal number of agents in the A assignment queue (and respectively for K_β). The A assignment queue is full when it holds K_α agents (and respectively K_β agents for B).

Definition 11. A buffer-queue mechanism $(\mathcal{M}, K_\alpha, K_\beta)$ is given by two assignment queues with arbitrary queueing policies and maximal size K_α, K_β, such that:

- If the item's assignment queue is not empty, the arriving item x_t is assigned to an agent on the assignment queue.
- If the item's assignment queue is empty, the mechanism asks a new agent to report her type. If the agent reports a mismatched type and the opposite assignment queue is not full, the agent is put on the queue and the mechanism continues to ask another agent. Otherwise, the agent is assigned the current item and the period ends.

The assignment queue serves as a buffer queue. When the mechanism gets an item it starts searching for a matching agent. If the assignment queue for the item is empty the

mechanism approaches new agents from the offer queue one by one, asking if they are a matching agent. When the mechanism approaches a mismatched agent the mechanism avoids misallocation by putting the agent in the assignment queue buffer and continuing the search for a matching agent. When the assignment queue is full, so that it is not incentive compatible for an agent to decline a mismatched item and join the assignment queue, the mechanism is forced to allocate the item to the next agent, who is mismatched with probability given by the prior. Misallocation only occurs when the assignment queue is full. Thus, the probability of mismatch can be derived from the probability that the assignment queue buffer is full.

In the previous section we essentially proved the following result:

Theorem 12. *The probability of misallocation in a* $(\mathcal{M}, K_\alpha, K_\beta)$ *mechanism is*

$$\xi^{\mathcal{M}} = \frac{2p(1-p)}{(1-p)K_\alpha + pK_\beta + 1}$$

Given theorem 12 we reduce the problem of finding the mechanism that maximizes welfare to the problem of finding the policy that gives maximal K_α, K_β. The limit on the size K_α, K_β is the incentive compatibility constraint of the agents. In section 3 we have shown how K_α, K_β are calculated from the agent's maximization problem given a policy for the assignment queue. In this section we proceed to find the incentive compatible policy that would give us the maximal K_α, K_β.

Notice that we can choose K_α independently of K_β. The constraint on K_α is the incentive compatibility constraint, which we formally define below. It requires that α agents prefer to join the A assignment queue rather than be immediately mismatched with a B item. The value of joining the assignment queue depends on the stochastic evolution of the A assignment queue that determines the expected delay until the agent is assigned her matching item. The policy for the B assignment queue will be in effect

only once the A assignment queue is empty, so the policy of the B assignment queue does not affect the expected wait of agents in the A queue. Therefore the incentives of agents to join the A assignment queue are independent of the policy for the B assignment queue and we can analyze them independently. Hence, we can find the optimal mechanism by finding the optimal queue policy for each assignment queue separately.

We consider a class of queueing policies to allow for general assignment policies – an arriving item will not necessarily be offered to the first agent in the queue. Instead of determining the size of the queue from the agent's decision to join, we specify a size for the queue and ask whether it is incentive compatible for agents to join.

Definition 13. A $\langle K, \varphi \rangle$ queuing policy is given by a maximal size of the queue $K \in \mathbb{N}$ and non-negative assignment probabilities $\varphi = \{\varphi(k,i)\}_{1 \leq i \leq k \leq K}$ such that $\sum_{i=1}^{k} \varphi(k,i) = 1$. If an item arrives when there are $1 \leq k \leq K$ agents in the queue it will be allocated to the agent in position i with probability $\varphi(k,i)$. Agents join the end of the queue and move a position forward if an agent in front of them was assigned.

This class of policies includes and extends common queueing policies. By setting

$$\varphi(k,i) = \begin{cases} 1 & i = 1 \\ 0 & i > 1 \end{cases}$$

we get the familiar First In First Out ($FIFO$) queueing policy with the queue length limited to K. The policy we analyzed in section 3 is equivalent to $FIFO$ policies for both assignment queues with K determined by the agents' decision. By setting

$$\varphi(k,i) = \begin{cases} 1 & i = k \\ 0 & i < k \end{cases}$$

we get the Last In First Out ($LIFO$) queueing policy.

Before we turn to give the conditions required for incentive compatibility of a $\langle K, \varphi \rangle$ policy we describe the value of taking a position in the queue. To simplify notation we consider the policy for the A assignment queue, that is A is the matching item, B is the mismatched item and α agents can join the assignment buffer-queue instead of being mismatched.

A position in the A assignment queue gives the α agent a stochastic distribution over the waiting time until she receives a matching A item. As in the previous section, we describe the stochastic transition using the state space $\hat{S}_\alpha = \{0, K\} \times \{\phi, B\}$, which is a restriction of the full state space of the mechanism. (k, B) denotes the state in which there are k agents of type α in the queue and the system is searching for a β agent to be assigned the current B item. The α agents that the mechanism encounters during the search will join the queue. The period ends when the current item is allocated, which happens either when an approached agent is a β or when the queue is full and the item is allocated to the next agent, α or β. When the item is allocated the system moves to a state (k, ϕ), which represents that there is no item to allocate and k is the number of agents that ended up in the queue. The system moves from a (k, ϕ) state by receiving an item. A B item moves the system to search for a β agent, represented by a move to (k, B). An A item will be allocated to an agent in the queue, according to $\varphi(k, \cdot)$ and the system moves to $(k - 1, \phi)$.

We denote the expected value to an agent who is in position i in the queue when the system is in state (k, ϕ) (i.e. there are k agents in the assignment queue) by $g(k, i)$, and the expected value to an agent who is in position i in the queue when the system is in state (k, B) (i.e. there are k agents in the assignment queue and the mechanism is making offers) by $g^B(k, i)$. The valuations are given as the solution to the dynamic programing equations:

$$
\begin{aligned}
g(k,i) &= p_B \cdot g^B(k,i) + p_A \cdot \varphi(k,i) \cdot 1 + \\
&\quad + p_A \cdot \left(\sum_{j=1}^{i-1} \varphi(k,j) \right) (g(k-1,i-1) - c) \\
&\quad + p_A \cdot \left(\sum_{j=i+1}^{k} \varphi(k,j) \right) (g(k-1,i) - c) \\
g^B(k,i) &= \begin{cases} p_\beta \cdot (g(k,i) - c) + p_\alpha \cdot g^B(k+1,i) & k < K \\ g(k,i) - c & k = K \end{cases}
\end{aligned}
$$

When a B item arrives the agent in position i stays in the same position, but other agents may join the queue. When an A item arrives there are three options. With probability $\varphi(k,i)$ the agent gets assigned the item, getting a value of 1 and no wait. If an agent $j \leq i$ gets assigned the item, the agent in position i moves to position $i-1$ out of a queue of $k-1$ agents. If an agent $j > i$ gets assigned the item, the agent in position i stays in position i but out of $k-1$ agents. When another agent is assigned a period passes and the waiting agent incurs a cost of waiting c.

Assuming that agents have correct beliefs over the evolution of the system, an agent's valuation for joining a queue of length $k-1$ is given by $g^B(k,k)$. The agent always joins the last position in the queue, so if there were $k-1$ agents in the queue before she joined she can take the k-th position of k agents. After the agent joins the queue the system will proceed to search for another β agent to take the item, and the agent takes into account the possibility that other agents may join the queue after her before the end of the period.

The following incentive compatibility constraint is represents the requirement that agents choose to join the queue (for every position) instead of taking an immediate mismatch:

Definition 14. A $\langle K, \varphi \rangle$ queueing policy is incentive compatible if every position $1 \leq$

$k \leq K$ in the queue is **acceptable**, that is

$$g^B(k,k) \geq v$$

The mechanism designer's problem is thus reduced to finding the incentive compatible $\langle K, \varphi \rangle$ policy for which K is maximal. To directly check that a policy is incentive compatible one has to solve the dynamic equations to determine $g^B(k,k)$ and verify that the IC constraint is satisfied for each k. In the following we adopt a simpler approach by first providing a bound on K using the *expected* IC constraint: If an agent is willing to join the queue for each position she must be also willing to join a random position in the queue.

Proposition 15. *Suppose that $p_\alpha = p_A = p$. There is no incentive compatible $\langle K, \varphi \rangle$ policy such that $K > K^*$, where $K^* = \left\lfloor 2p\frac{1-v}{c} \right\rfloor - 1$.*

To prove this, we use the following simplification. We decompose the utility of taking a position k as $g^B(k,k) = 1 - c \times w_k$, where w_k is the expected waiting time for an agent who joins the queue at position k. The condition for incentive compatibility is equivalent to $w_k \leq \frac{1-v}{c}$ for every w_k. We can calculate each w_k from K and φ by solving the dynamic programing equations, but we can get a much simpler expression by looking at $\mathbb{E}[w_k]$, the expected waiting time for an agent who enters the queue at a random position.

Lemma 16. *Suppose that $p_\alpha = p_A = p$ then we have that*

$$\mathbb{E}[w_k] = \frac{K+1}{2p}$$

independently of φ.

Proof. Restrict attention to the time when the queue is not empty. Notice that the choice of φ does not affect the number of agents in the queue, and we can use the calculations

from the proof of lemma7 to get the stationary distribution of the number of agents in the queue. When $p_A = p_\alpha = p$ the stationary distribution of the number of agents in the queue is

$$\hat{\pi}(k) = \frac{1}{K}$$

By Little's Law the expected waiting time for a randomly chosen α agent in the system is equal to the expected number of agents waiting divided by the arrival rate at which new α agents join the queue, which is equal to the rate p_A at which α agents get assigned and leave the queue. Therefore we have:

$$\mathbb{E}[w_k] = \sum_{k=1}^{K} k \cdot \hat{\pi}(k)/p = \frac{K(K+1)}{2}\frac{1}{K}/p = \frac{K+1}{2p}$$

□

Using this lemma we can prove our result:

Proof. Consider an incentive compatible $\langle K, \varphi \rangle$ queue with maximal size K. From the IC constraints we have that for every k

$$w_k \leq \frac{1-v}{c}$$

The IC constraint must also hold in expectation:

$$\mathbb{E}[w_k] \leq \frac{1-v}{c}$$

so

$$\frac{K+1}{2p} \leq \frac{1-v}{c}$$

or

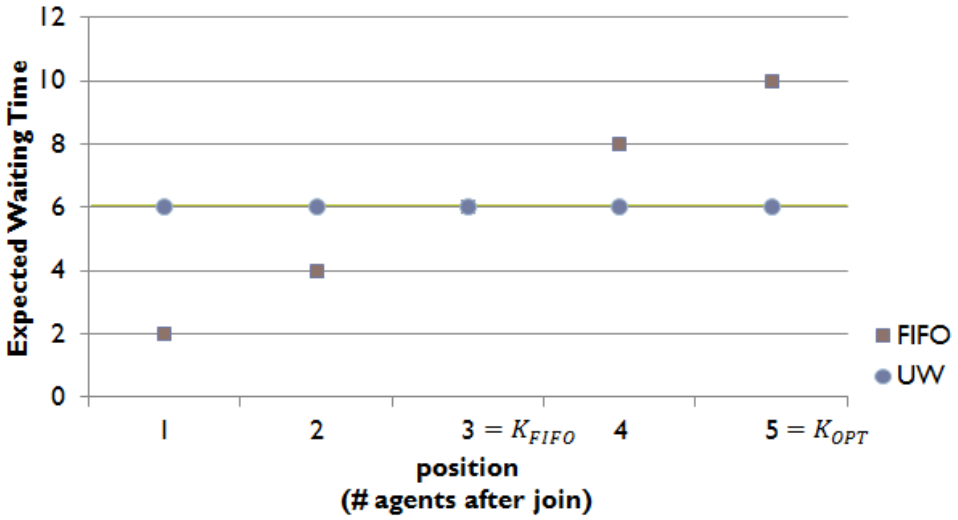


Figure 4.1: Expected waiting time per position w_k under $FIFO$ and UW for $K = 5, p = \frac{1}{2}$. The green line denotes $\frac{1-v}{c} = 6 = \frac{K+1}{2p}$.

$$K \leq \left\lfloor 2p\frac{1-v}{c} \right\rfloor - 1$$

yielding the required result. □

Can we achieve an incentive compatible queue with $K = K^*$? Suppose that $K^* = 2p\frac{1-v}{c} - 1$ is an integer, and consider a $\langle K^*, \varphi \rangle$ policy. From the lemma we know that $\mathbb{E}[w_k] = \frac{1-v}{c}$. For the IC to hold for every k we must have $w_k \leq \frac{1-v}{c}$, so we must have that for each k the expected wait is $w_k = \frac{1-v}{c}$.

Definition 17. The K uniform wait (UW_K) queue is a $\langle K, \varphi \rangle$ policy where the expected wait for an agent who joins the last position is independent of the number agents already on the queue, that is, an agent that joins the queue in position $1 \leq k \leq K$ faces the same expected wait of $w_k = \frac{K+1}{2p}$.

Theorem 18. *Setting the policy of the assignment queues to be $UW[K^*_\alpha]$ and $UW[K^*_\beta]$*

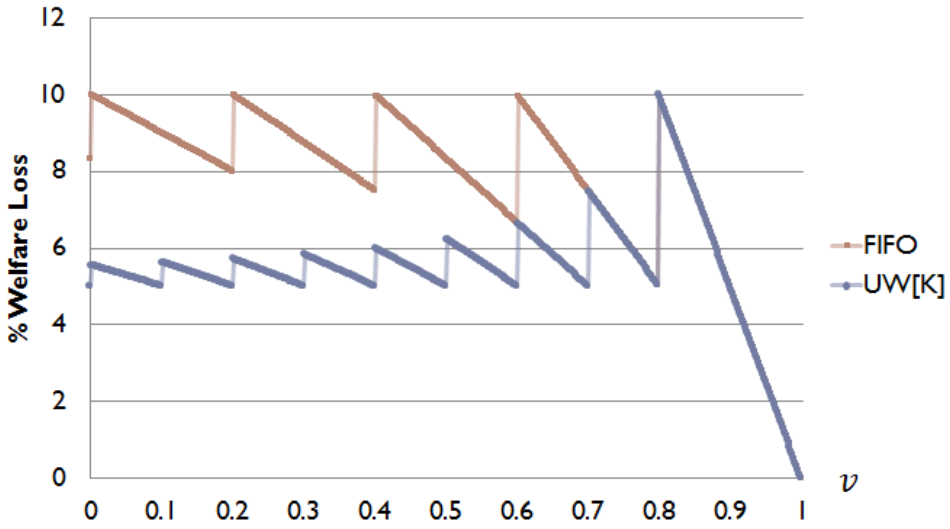


Figure 4.2: Welfare loss under the $FIFO$ and UW policies. Parameters: $c = 0.1, p = \frac{1}{2}$.

for $K^*_\alpha = \left\lfloor 2p\frac{1-v}{c} \right\rfloor - 1$, $K^*_\beta = \left\lfloor 2(1-p)\frac{1-v}{c} \right\rfloor - 1$ gives the minimal misallocation probability

$$\xi^{OPT} = \frac{2p(1-p)}{(1-p)K^*_\alpha + pK^*_\beta + 1}$$

and highest welfare out of all incentive compatible queue-based policies.

4.1 Designing a Uniform Wait Queue

How can we achieve a uniform wait queue? Consider the $FIFO$ (first in first out) queue with a limit size of $K = 2p\frac{1-v}{c} - 1$, which is a $\langle K, \varphi \rangle$ policy where $\varphi(k,i) = \begin{cases} 1 & i = 1 \\ 0 & i > 1 \end{cases}$, that is the item is always assigned to the first agent in the queue. The expected waiting time for an agent joining the queue at position k is $w_k = \frac{k}{p}$, and the vector of normalized waiting times $p \cdot \vec{w} = (pw_1, \ldots, pw_K)$ is:

$$p \cdot \vec{w}^{FIFO} = (1, 2, 3, \ldots, K)$$

Under this queuing policy the agents who join the last positions in the queue face a longer waiting time than the average waiting time $\frac{K+1}{2p} = \frac{1-v}{c}$, violating their IC constraint. The agents who join the first positions in the queue face a waiting time that is strictly lower than average, and their IC constraint is slack.

Consider the $LIFO$ (last in first out) queue with a limit size of $K = 2p\frac{1-v}{c} - 1$, which is a $\langle K, \varphi \rangle$ policy where $\varphi(k,i) = \begin{cases} 1 & i = k \\ 0 & i < k \end{cases}$, that is, the item is always assigned to the last agent in the queue. The expected waiting time for an agent joining the queue at position k is $w_k = \frac{K-k+1}{p}$, and the vector of normalized waiting times is:

$$p \cdot \vec{w}^{LIFO} = (K, K-1, \ldots, 3, 2, 1)$$

Under $LIFO$ the agents who join the first positions in the queue face a longer waiting time than the average waiting time $\frac{K+1}{2p} = \frac{1-v}{c}$, violating their IC constraint. The agents who join the last positions in the queue face a waiting time that is strictly lower than average, and their IC constraint is slack.

To satisfy the IC constraints of all agents we would like to have an "intermediate" policy, which gives agents a waiting time between $FIFO$ and $LIFO$. We can calculate the uniform wait policy by exploiting the fact that agents only move forward in the queue. Since the first position never moves up w_k is determined only by $\varphi(k,1)$. We will sequentially choose $\varphi(k,i)$ to ensure that $w_k = \frac{K+1}{2p}$.

The following algorithm can calculate such an intermediate policy that would give a UW queue. We start by setting $\varphi_0(k,1) = 1$ for all $k \leq K$, giving us the $FIFO$ policy

$$\varphi_0 = \begin{pmatrix} 1 & & & & \\ 1 & 0 & & & \\ 1 & 0 & 0 & & \\ 1 & 0 & 0 & 0 & \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Under this queueing policy we have $p \cdot w_1 = 1 < \mathbb{E}[p \cdot w_k] = \frac{K+1}{2}$. We vary φ continuously by choosing $x \in [2, K+1]$ and setting

$$\begin{aligned} \varphi_1(k,1)\{x\} &= \begin{cases} 1 & k+1 < x \\ 0 & x < k-1 \\ x-k & o/w \end{cases} \\ \varphi_1(k,2)\{x\} &= 1 - \varphi(k,1) \end{aligned}$$

Notice that w_1 depends only on the values of $\{\varphi(k,1)\}_{k=1}^{K}$. For $x = K$ we get the $FIFO$ queue policy and $p \cdot w_1 = 1$. For $x = 1$ that we get that $\{\varphi_1(k,1)\{x\}\}_{k=1}^{K}$ have the same values as in the $LIFO$ queue policy, implying that $p \cdot w_1 = K$. Since w_1 varies continuously with x there is some $x_1 \in [1, K]$ such that $w_1 = \frac{K+1}{2}$. The resulting φ_1 will look something like this:

$$\varphi_1 = \begin{pmatrix} 1 & & & & \\ 1 & 0 & & & \\ x_1 & 1-x_1 & 0 & & \\ 0 & 1 & 0 & 0 & \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

If under the updated φ we still have that $p \cdot w_2 < \frac{K+1}{2}$ we start increasing w_2 by shifting

probability from $\varphi(k,2)$ to $\varphi(k,3)$ in an analogous way to the previous stage. Let x_2 be the value for which $p \cdot w_2 = \frac{K+1}{2}$, and we find that

$$\varphi_2 = \begin{pmatrix} 1 & & & & \\ 1 & 0 & & & \\ x_1 & 1-x_1 & 0 & & \\ 0 & x_2 & 1-x_2 & 0 & \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Notice that we still have $p \cdot w_1 = \frac{K+1}{2}$ since we didn't change any of the $\varphi(k,1)$. Continuing in an analogous manner, we obtain w_3, w_4 and the full $\varphi^{UW[5]}$.

Example 19. The $UW[2]$ queue policy is a $\langle K, \varphi \rangle$ policy with $K = 2$ and

$$\varphi = \begin{pmatrix} 1 & \\ \frac{1}{3-p} & \frac{2-p}{3-p} \end{pmatrix}_{k \times i}$$

and has an expected wait of $w_k = 1.5/p$ for all k.

Example 20. The $UW[3]$ queue policy is a $\langle K, \varphi \rangle$ policy with $K = 3$ and

$$\varphi = \begin{pmatrix} 1 & & \\ \frac{1}{2-p} & \frac{1-p}{2-p} & \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}_{k \times i}$$

and has an expected wait of $w_k = 2/p$ for all k.

Notice that to implement a uniform wait queue the designer needs to know the parameters of the environment. The assignment probabilities of $UW[3]$ do not match the assignment probabilities of $UW[2]$ in the overlapping range $k \leq 2$. The designer needs to select a queue size K and implement the assignment probabilities for that specific K.

Even for a fixed K, the assignment probabilities depend on the arrival probability p.

Under the UW policy an agent's decision to join the assignment queue depends on her belief about the decisions of agents that follow her in the queue. Under the $FIFO$ queue policy the waiting time of an agent is independent of the number of agents behind her in the queue, but under UW the policy shifts assignment probabilities when more agents join the queue. For example, an agent joining the first position in the $UW[2]$ queue knows that the mechanism will make an offer to the agent behind her in the offer queue. If the agent behind her accepts the B item, leaving her alone in the assignment queue, she will have a conditional expected wait of $\frac{3}{2p} - \frac{1}{2}$, which is shorter than the average wait $\frac{3}{2p}$. To balance her expected wait, she has a longer conditional expected wait of $\frac{3}{2p} + \frac{1-p}{2p}$ if the agent behind her joins the queue. An agent with rational expectation will join the queue, given the expected waiting time of $p \times \left(\frac{3}{2p} - \frac{1}{2}\right) + (1-p) \times \left(\frac{3}{2p} + \frac{1-p}{2p}\right) = \frac{3}{2p}$, even though she will regret this decision if the agent behind her ends up joining the queue as well.

When the mechanism is the UW policy for the correct parameters p and $\frac{1-v}{c}$ and all agents have correct beliefs the agents' decision is simple—they join the assignment queue whenever it is not full. However, this heavily depends on having the right parameter specification. If the mechanism is a UW for misspecified parameters the agents play a game—the decision whether to join the queue at position k depends on the probability that the following agents will join the queue.

5 Robust design

The aim of this section is to present a simpler policy, service in random order ($SIRO$), and show that it is the optimal policy in a "robust" sense. We aim to address two concerns about the uniform wait (UW) mechanism, which we derived in the previous section. First, to implement the UW mechanism the designer must know the precise parameters of the environment . Second, the UW policy is incentive compatible for agents only under the

assumption that agents hold the exact specified probabilistic beliefs.

The following example illustrates that a UW policy tailored for a certain environment performs poorly in different, albeit similar, environments.

Example 21. Assume that $p = 1/2$, making the A and B assignment queues symmetric. Consider the mechanism that implements a queueing policy $UW[4]$ for both assignment queues, given by $\left\langle 4, \varphi^{UW[4]} \right\rangle$ with

$$\varphi^{UW[4]} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{6}{7} & \frac{1}{7} & 0 & 0 \\ 0 & \frac{4}{5} & \frac{1}{5} & 0 \\ 0 & 0 & \frac{6}{11} & \frac{5}{11} \end{pmatrix}$$

This policy is optimal when $p = 1/2$ and agents are willing to wait up to $\frac{1-v}{p\,c} = 5$ periods, giving a misallocation probability of $\frac{1}{10}$. Suppose that the designer was slightly wrong and $\frac{1-v}{c} = 4.5$, so the $UW[4]$ policy is not incentive compatible. Under this policy the agents play a game, where the possible strategies are whether to join each one of the potential queue positions $1, 2, 3, 4$. The unique symmetric equilibrium of this game is that agents join positions $1, 2$, but accept a mismatched item instead of joining positions $3, 4$. Thus the policy $UW[4]$ would yield a buffer of size $K = 2$ and a misallocation probability of $\frac{1}{6}$, which is the same as a $FIFO$ policy would yield. A $UW[3]$ policy could have given a buffer size of $K = 3$ and misallocation probability of $\frac{1}{8}$.

Note that preventing agents from joining position 4 on the $UW[4]$ queue does not produce a $UW[3]$ queue, as the two policies have different assignment probabilities $\varphi(k, i)$ for $k \leq 3$.

The following example shows that UW can perform worse than $FIFO$, if used for misspecified parameters.

Example 22. Consider a designer that, mistakenly, believes that $p = 9/10$ and that agents are willing to wait up to $\frac{1-v}{pc} = 3\frac{1}{3}$. The $UW[5]$ for the A assignment queue under these parameters is given by $\left\langle 5, \varphi^{UW[5]} \right\rangle$ with

$$\varphi^{UW[5]} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ \frac{40}{61} & \frac{21}{61} & 0 & 0 & 0 \\ 0 & \frac{90}{241} & \frac{35860}{65311} & \frac{21}{271} & 0 \\ 0 & 0 & 0 & \frac{20}{39} & \frac{19}{39} \end{pmatrix}$$

This queueing policy is optimal given these beliefs, but assume that the correct probability is $p = 2/3$ and $\frac{1-v}{c} = 4.5$. Under these parameters it is not incentive compatible for agents to join all positions of the $UW[5]$ queue. It is a symmetric trembling hand perfect equilibrium for agents to join the queue when offered positions $1, 2, 4, 5$, while accepting the mismatched item when offered position 3. Thus the $UW[5]$ would have a maximal buffer-queue size of 2, whereas the $FIFO$ policy would yield a maximal buffer-queue size of 3.

These two examples show that in order to use the UW policy the designer must know the correct parameters. But even if the designer knows the correct parameters, the optimal mechanism will be incentive compatibility only if the agents have correct beliefs, as the expected wait in the UW queue depends on whether other agents will join the queue as well. Under UW an agent that joins position 1 will have a shorter than average wait if an item arrives before other agents join. To balance this and ensure that $w_1 = \frac{K+1}{2p}$, the mechanism gives the agent in position 1 have a longer than average wait when other agents join before an item arrives. A pessimistic agent who overstates the probability that other agents will join the queue after her will thus be unwilling to join.

The $FIFO$ policy is simple, parameter free and does not rely on agents holding correct

beliefs. However, as we saw in the previous section, the $FIFO$ policy is inefficient, as the waiting time is spread unequally between the different positions. The Service In Random Order ($SIRO$) policy, which can be implemented by $\varphi(k,i) = \frac{1}{k}$, is a simple and parameter free policy policy as well. In this section we show that $SIRO$ outperforms the $FIFO$ policy, and show that $SIRO$ is scalable-optimal (as we define below).

We start by considering α agents who hold general beliefs. We assume that agents have correct beliefs on the item arrival probabilities, i.e. agents know the true probabilities p_A, p_B. But following Wilson (1987); Bergemann and Morris (2005), we allow the agents to hold arbitrary beliefs about the types of agents that are behind them in the offer queue. We denote by $\sigma(a)$ the probability the agents assigns to agent $a = 1, 2, 3...$ being of type α, where $a = 1$ is the immediately successor in the offer queue[20] and so on. The belief σ is not restricted to be i.i.d., as we allow $\sigma(1) \neq \sigma(2)$. An agent decides whether to join the assignment queue at a certain position based on the valuation of the position given her beliefs σ at the time she is offered an item. Given a belief σ the subjective value of being in position i out of k agents in the queue when the next agent in the offer queue is a is denoted by $g_{\sigma,a}(k,i)$ and similarly $g^B_{\sigma,a}(k,i)$ when the system is still searching to allocate a B item. Note the two added subscripts. The value depends on the belief about future joining rates σ. Since we consider joining rates that are no longer stationary, the value of positions is no longer stationary. The subscript a denotes the next agent who will be offered to join the assignment queue, tracking the position in the offer queue reached.

Given a $\langle K, \varphi \rangle$ policy the valuations are the solution to the dynamic programing equations:

[20] Note that all agents have the same belief *at the time they are being offered an item.*

$$
\begin{aligned}
g_{\sigma,a}(k,i) &= p_B \cdot g^B_{\sigma,a}(k,i) + p_A \cdot \varphi(k,i) \cdot 1 + \\
&\quad + p_A \cdot \left(\sum_{j=1}^{i-1} \varphi(k,j) \right) (g_{a,\sigma}(k-1,i-1) - c) \\
&\quad + p_A \cdot \left(\sum_{j=i+1}^{k} \varphi(k,j) \right) (g_{a,\sigma}(k-1,i) - c) \\
g^B_{\sigma,a}(k,i) &= \begin{cases} (1-\sigma(a)) \cdot (g_{a,\sigma}(k,i) - c) + \sigma(a) \cdot g^B_{a+1,\sigma}(k+1,i) & k < K \\ g_{a,\sigma}(k,i) - c & k = K \end{cases}
\end{aligned}
$$

Note that for any sequence $\sigma(\cdot)$ the values $g^B_{\sigma,a}(k,i), g_{\sigma,a}(k,i)$ are well defined[21]. To simplify notation we denote $g^B_\sigma(k,i) = g^B_{\sigma,1}(k,i)$. Using this notation we form a robust incentive constraint:

Definition 23. A $\langle K, \varphi \rangle$ policy is Belief Free Incentive Compatible (Bf-IC) if for all positions $1 \leq k \leq K$ and all beliefs σ we have

$$g^B_\sigma(k,k) \geq v$$

By requiring Bf-IC instead of IC we make the mechanism "parameter free" for the participating agents.

The $UW[K^*]$ policy does not satisfy Bf-IC . For illustration, suppose that $K^* = 2$. As we saw in the previous section, agents are willing to join the first position in the $UW[2]$ queue because they are willing to accept a lottery between a short waiting time if they are alone in the queue and a longer waiting time if another agent joins the queue. If an

[21]The mechanism can make offers to at most K agents from the offer queue before drawing another object. When the mechanism draws an object, there is a positive and constant probability that the next K items are all matching items and all agents in the assignment queue will be assigned. Therefore, the expected wait must be finite for every σ. Furthermore, this implies that the probability that agent a is made an offer before the current agent is assigned decreases at an exponential rate as $a \to \infty$. Combining, we find that $g^B_{\sigma,a}(k,i)$ is well defined.

agent is offered position 1 and she believes that $\sigma(1) = 1$, that is that the agent behind her is an α agent who will join the assignment queue, the agent will evaluate this offer as being offered position $i = 1$ out of $k = 2$ agents in the queue. Under the $UW[2]$ policy, even under the true beliefs $g^B(2,1) < v$. So under the beliefs $\sigma(a) = \begin{cases} 1 & a = 1 \\ p_\alpha & a > 1 \end{cases}$ the agent will mismatch instead of joining the first position.

Definition 24. A mechanism is weakly regret free if for every k, i and belief σ position i out of k is acceptable, that is:

$$g^B_\sigma(k,i) \geq v$$

For the queue policy to be Bf-IC it must be that every position in the queue is acceptable.

Lemma 25. *A $\langle K, \varphi \rangle$ policy is Bf-IC if and only if it is weakly regret free.*

Proof. If all positions are acceptable then in particular the policy Bf-IC. To prove the opposite direction, let $\langle K, \varphi \rangle$ be a Bf-IC policy, and consider any belief σ and position k, i. Construct a new belief $\sigma'(a) = \begin{cases} 1 & a \leq k - i \\ \sigma\left(a - (k-i)\right) & a > k - i \end{cases}$. From the Bf-IC condition we have that $v \leq g^B_{\sigma'}(i,i) = g^B_\sigma(k,i)$. □

Remark 26. The $FIFO$ policy satisfies Bf-IC whenever it satisfies IC, as the value of a position in the $FIFO$ queue is independent of the number of agents in the queue.

We now turn to relax the information required from the mechanism designer. We ask for a "one size fits all" mechanism, which does not depend on the parameters of the situation. A *scalable* mechanism is a specification of the buffer-queue policy for any number of agents in the queue. The mechanism is allowed to tailor the mechanism to the parameters of the environment only by restricting access to the buffer-queue.

Definition 27. A scalable queue policy is $(\langle\varphi\rangle, K_\varphi(v))$, where $\varphi = \{\varphi(k,i)\}_{1\le i\le k<\infty}$ specifies assignment probabilities for any length of the queue such that $\frac{\varphi(k,i)}{\varphi(k,j)} = \frac{\varphi(k',i)}{\varphi(k,j)}$ whenever $i,j \le \min\{k,k'\}$, and $K_\varphi(v)$ specifies the last position at which agents are allowed to join.

Definition 28. A scalable queue policy $(\langle\varphi\rangle, K_\varphi(v))$ is Bf-IC if for every v the policy $\langle K_\varphi(v), \varphi\rangle$ is Bf-IC.

A scalable mechanism can be also viewed as a series of mechanisms $\{\langle K, \varphi_K\rangle\}_{K=1}^{\infty}$ such that $\varphi_K(k,i) = \varphi_{K'}(k,i)$ for any $k,i \le \min(K,K')$. From examples 19 and 20 we can see that the sequence $\left\{\left\langle K, \varphi^{UW[K]}\right\rangle\right\}_{K=1}^{\infty}$ is not a scalable queue policy. In contrast, the $FIFO$ policy is scalable:

Example 29. $FIFO$ is a scalable Bf-IC policy with $\varphi(k,i) = \begin{cases} 1 & i=1 \\ 0 & i>1 \end{cases}$ and $K_\varphi(v) = \left\lfloor p_A \frac{1-v}{c} \right\rfloor$.

The designer would like to select a scalable policy that would have the maximal the queue length $K_\varphi(v)$, but the designer does not know v. Given a prior over the relative probabilities of $v = v_1, v_2$ the designer can compare policies φ, φ' such that $K_\varphi(v_1) > K_{\varphi'}(v_1)$ but $K_\varphi(v_2) < K_{\varphi'}(v_2)$, by weighing the benefits and the probabilities of v_1, v_2. Without a prior there is no clear comparison between the two policies. Different policies are optimal for different values of v, as there is no scalable policy that would perform as well as $UW[2]$ when the degenerate prior is $v = 1 - c \times \frac{3}{2p}$ and as well as $UW[3]$ when the degenerate prior is $v = 1 - c \times \frac{2}{p}$. But while the mechanism designer may not know the distribution of v, we do know that the misallocation probability decreases in K with a decreasing marginal effect. That is, the gain from increasing the queue size from $K = 1$ to $K = 2$ is greater than the gain from increasing the queue size from $K = 11$ to $K = 12$. If the designer does not have any evidence suggesting that v_1 is more probable than v_2

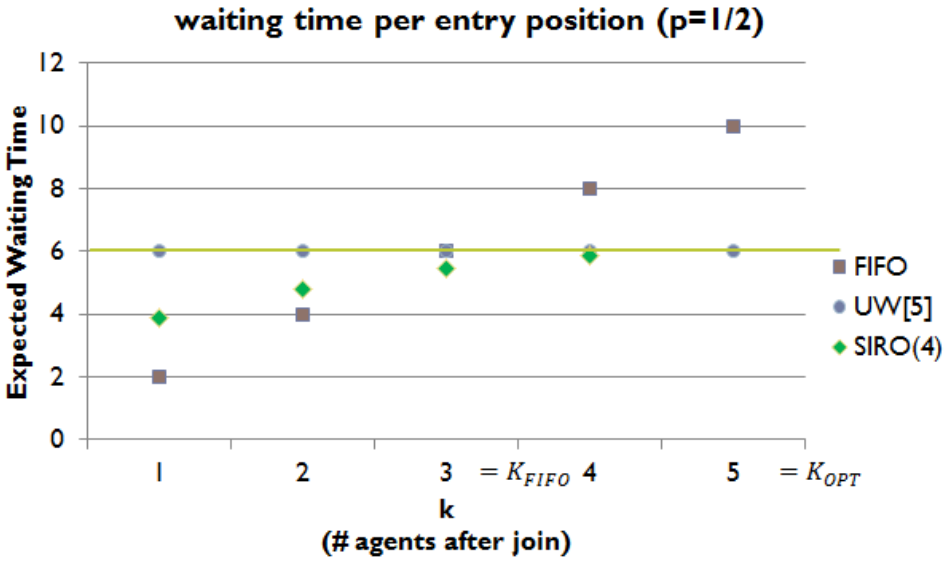


Figure 5.1: Expected waiting time per position w_k under $FIFO$, $UW[5]$ and $SIRO[4]$ for $p = \frac{1}{2}$. The green line denotes $\frac{1-v}{c} = 6 = \frac{K+1}{2p}$. Under the $SIRO$ policy the maximal buffer size will be 4, compared to 5 under UW and 3 under $FIFO$.

or vice versa he should choose the policy φ that maximizes $\min\{K_\varphi(v_1), K_\varphi(v_2)\}$. We therefore set up the following optimality condition:

Definition 30. A policy $(\langle\varphi\rangle, K_\varphi(v))$ is scalable-optimal if it is scalable Bf-IC and there is no other scalable Bf-IC policy $\left(\langle\varphi'\rangle, K'_{\varphi'}(v)\right)$ and $\bar{K}$ such that for all $v \leq 1$

$$\min\left\{K'_{\varphi'}(v), \bar{K}\right\} \geq \min\left\{K_\varphi(v), \bar{K}\right\}$$

with strict inequality for some v.[22]

In other words, a scalable-optimal mechanism requires the minimal amount of patience from agents to get a queue of size K conditional on requiring the minimal amount of

[22] A scalable-optimal strategy is a winning strategy for player 1 in the following game:

1. Player 1 chooses a scalable Bf-IC $(\langle\varphi\rangle, K_\varphi(v))$
2. Player 2 chooses a scalable Bf-IC $(\langle\varphi'\rangle, K_{\varphi'}(v))$ and a v_1
3. Player 1 chooses v_2

Player 2 wins if $\min\{K_{\varphi'}(v_1), K_{\varphi'}(v_2)\} \geq \min\{K_\varphi(v_1), K_\varphi(v_2)\}$ and $K_{\varphi'}(v_1) > K_\varphi(v_1)$, otherwise player 1 wins.

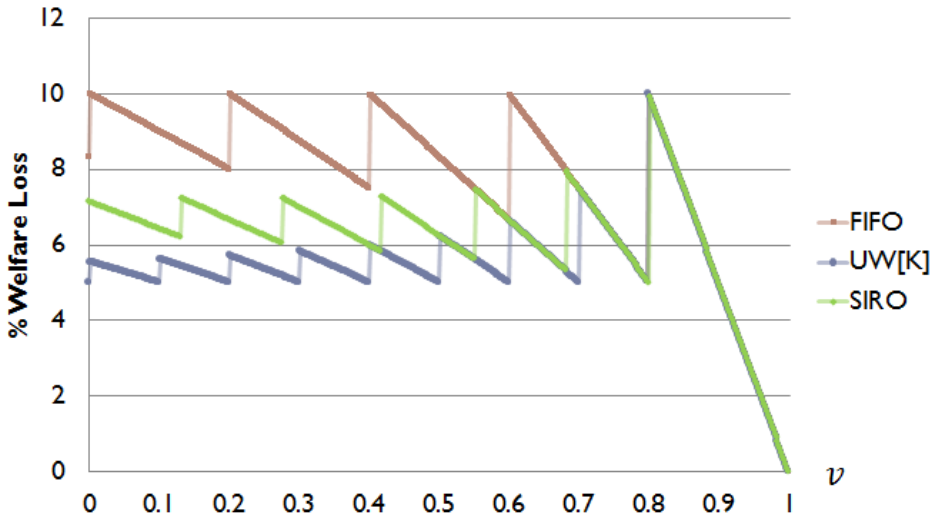


Figure 5.2: Welfare loss under the $FIFO$,UW and $SIRO$ policies. Parameters: $c = 0.1, p = \frac{1}{2}$.

patience for $1, \ldots, K-1$. We find that the unique scalable-optimal policy is a simple queueing policy:

Theorem 31. *The unique scalable-optimal policy is the Service In Random Order ($SIRO$) queue policy, in which $\varphi(k,i) = \frac{1}{k}$ for all k.*

6 Conclusion

This paper presents an analysis of the allocative efficiency of waiting lists. In congested systems where agents have homogenous waiting costs, the allocative efficiency depends only on the probability of misallocation. I consider a class of dynamic direct-revelation buffer-queue policies and show that common policies can be represented as mechanisms in this class. Welfare under a buffer-queue mechanism is shown to have a simple form: the size of the buffer-queue, i.e. the number of agents waiting for their preferred item that the mechanism can simultaneously hold, determines the probability of misallocation. When the buffer-queue is full the mechanism is forced to allocate items at random, as it cannot

incentivize agents to decline mismatched items. Maximizing the size of the buffer-queue will therefore minimize the probability of misallocation. For a given policy, the size of the buffer-queue is determined by the incentive compatibility constraint—agents are willing to decline a mismatched item only if their wait for a preferred item is below a threshold, but the mechanism can offer only a limited number of agents their preferred items within a short wait.

Using this analysis I am able to derive the optimal policy. The size of buffer-queue is maximized by equalizing the waiting time of agents who join different positions in the buffer-queue. The Uniform Wait (UW) queueing policy is introduced and shown to give the maximal welfare. While the UW policy is optimal, its performance is sensitive to the specification of the environment and the beliefs of the agents. The Service In Random Order (SIRO) policy, which gives all agents in the buffer-queue equal probability to of obtaining arriving item, is shown to be a robust and approximately optimal policy. The SIRO policy extends the size of the buffer-queue by offering higher incentives for the last agent to join, and achieves close to the optimal level of welfare.

Misallocation in waiting lists can cause substantial welfare losses. Allowing agents to express their preferences is essential to minimize misallocation, but it may not be enough. To safely allow agents to express their preferences, the mechanism should not penalize agents who decline items. This paper shows that simply allowing agents who decline items to keep their place in line greatly improves welfare and generates an approximately efficient allocation when agents are patient. When agents are impatient the SIRO policy generates greater welfare by making it safe for more agents to decline mismatched items, thus allowing more items to go to the agents who value them the most.

References

Abdulkadiroglu, A., and S. Loerscher (2007): "Dynamic house allocations," *Working papar*.

Adachi, H. (2003): "A search model of two-sided matching under nontransferable utility," *Journal of Economic Theory*, 113(2), 182–198.

Adan, I., and G. Weiss (2010): "Exact FCFS matching rates for two infinite multi-type sequences," *Submitted for publication*.

Alagoz, O., L. Maillart, A. Schaefer, and M. Roberts (2007): "Determining the acceptance of cadaveric livers using an implicit model of the waiting list," *Operations Research*, 55(1), 24–36.

Allon, G., A. Bassamboo, and I. Gurvich (2011): "We will be right with you: Managing customers with vague promises and cheap talk," *Oper. Res.*

Altman, E., and N. Shimkin (1998): "Individual equilibrium and learning in processor sharing systems," *Operations Research*, pp. 776–784.

Athey, S., and I. Segal (2007): "An efficient dynamic mechanism," *Unpublished manuscript, Harvard University*.

Babaioff, M., L. Blumrosen, and A. Roth (2010): "Auctions with online supply," in *Proceedings of the 11th ACM conference on Electronic commerce*, pp. 13–22. ACM.

Barzel, Y. (1974): "A theory of rationing by waiting," *Journal of Law and Economics*, 17(1), 73–95.

Bell, C., and S. Stidham Jr (1983): "Individual versus social optimization in the allocation of customers to alternative servers," *Management Science*, pp. 831–839.

Ben-Shahar, I., A. Orda, and N. Shimkin (2000): "Dynamic service sharing with heterogeneous preferences," *Queueing systems*, 35(1), 83–103.

Bergemann, D., and S. Morris (2005): "Robust mechanism design," *Econometrica*, 73(6), 1771–1813.

——— (2010): "Robust implementation in general mechanisms," *Games and Economic Behavior*.

Bergemann, D., S. Morris, and O. Tercieux (2011): "Rationalizable implementation," *Journal of Economic Theory*.

Bergemann, D., and M. Said (2010): "Dynamic auctions: A survey," *Cowles Foundation Discussion Paper No. 1757R*.

Bergemann, D., and J. Välimäki (2006): "Efficient dynamic auctions," *Cowles Foundation discussion papers*, 1584.

——— (2010): "The dynamic pivot mechanism," *Econometrica*, 78(2), 771–789.

Bloch, F., and D. Cantala (2008): "Markovian assignment rules," *Woring paper*.

Caldentey, R., E. Kaplan, and G. Weiss (2009): "FCFS infinite bipartite matching of servers and customers," *Advances in Applied Probability*, 41(3), 695–730.

Chen, H., and M. Frank (2001): "State dependent pricing with a queue," *IIE Transactions*, 33(10), 847–860.

Chung, K., and J. Ely (2003): "Implementation with Near-Complete Information," *Econometrica*, 71(3), 857–871.

Cullis, J., and P. Jones (1986): "Rationing by waiting lists: an implication," *The American Economic Review*, 76(1), 250–256.

David, I., and U. Yechiali (1995): "One-attribute sequential assignment match processes in discrete time," *Operations Research*, pp. 879–884.

Diamond, R. (2011): "The Welfare Implications of Changes in Local Wages, Prices, and Amenities across US Cities: 1980&2000.," .

Dizdar, D., A. Gershkov, and B. Moldovanu (2011): "Revenue maximization in the dynamic knapsack problem," *Theoretical Economics*, 6(2), 157–184.

Fuhrmann, S., and I. Iliadis (1994): "A comparison of three random disciplines," *Queueing systems*, 18(3), 249–271.

Gershkov, A., and B. Moldovanu (2010): "Efficient sequential assignment with incomplete information," *Games and Economic Behavior*, 68(1), 144–154.

Glaeser, E., and E. Luttmer (2003): "The Misallocation of Housing Under Rent Control," *The American Economic Review*, 93(4), 1027–1046.

Gujar, S., and D. Parkes (2010): "Dynamic Matching with a Fall-back Option," in *Proceeding of the 2010 conference on ECAI 2010: 19th European Conference on Artificial Intelligence*, pp. 263–268. Citeseer.

Hassin, R. (1986): "Consumer information in markets with random product quality: the case of queues and balking," *Econometrica: Journal of the Econometric Society*, pp. 1185–1195.

——— (2009): "Equilibrium customers choice between FCFS and random servers," *Queueing Systems*, 62(3), 243–254.

Hassin, R., and M. Haviv (2003): *To queue or not to queue: Equilibrium behavior in queueing systems*, vol. 59. Springer.

Hylland, A., and R. Zeckhauser (1979): "The efficient allocation of individuals to positions," *The Journal of Political Economy*, pp. 293–314.

Kakade, S., I. Lobel, and H. Nazerzadeh (2011): "Optimal dynamic mechanism design and the virtual pivot mechanism," *Unpublished manuscript, Microsoft Research New England*.

Kaplan, E. (1984): "Managing the demand for public housing," Ph.D. thesis, Massachusetts Institute of Technology, Dept. of Urban Studies and Planning.

——— (1986): "Tenant assignment models," *Operations research*, pp. 832–843.

——— (1987): "Tenant assignment policies with time-dependent priorities," *Socio-Economic Planning Sciences*, 21(5), 305–310.

——— (1988): "A public housing queue with reneging and task-specific servers," *Decision Sciences*, 19(2), 383–391.

Kennes, J., D. Monte, and N. Tumennasan (2011): "The Daycare Assignment Problem," *Economics Working Papers*.

Kurino, M. (2009): "House allocation with overlapping agents: A dynamic mechanism design approach," *Jena Economic Research Papers*.

Leslie, P., and A. Sorensen (2009): "The Welfare Effects of Ticket Resale," .

Lindsay, C., and B. Feigenbaum (1984): "Rationing by waiting lists," *The American economic review*, 74(3), 404–417.

Moulin, H. (2008): "Proportional scheduling, split-proofness, and merge-proofness," *Games and Economic Behavior*, 63(2), 567–587.

Moulin, H., and R. Stong (2002): "Fair queuing and other probabilistic allocation methods," *Mathematics of Operations Research*, pp. 1–30.

Naor, P. (1969): "The regulation of queue size by levying tolls," *Econometrica: journal of the Econometric Society*, pp. 15–24.

Nichols, D., E. Smolensky, and T. Tideman (1971): "Discrimination by waiting time in merit goods," *The American Economic Review*, 61(3), 312–323.

Niederle, M., and L. Yariv (2007): "Matching through decentralized markets," Discussion paper, mimeo, Stanford University.

Pai, M., and R. Vohra (2011): "Optimal dynamic auctions and simple index rules," *Unpublished manuscript, University of Pennsylvania.*

Parkes, D., R. Cavallo, F. Constantin, and S. Singh (2010): "Dynamic Incentive Mechanisms," *Ann Arbor*, 1001, 48109.

Parkes, D., and S. Singh (2003): "An MDP-based approach to online mechanism design," in *Proc. 17th Annual Conf. on Neural Information Processing Systems (NIPS03)*. Citeseer.

Pavan, A., I. Segal, and J. Toikka (2009): "Dynamic mechanism design: Incentive compatibility, profit maximization and information disclosure," .

Polterovich, V. (1993): "Rationing, queues, and black markets," *Econometrica: Journal of the Econometric Society*, pp. 1–28.

Roth, A. (1982): "The economics of matching: Stability and incentives," *Mathematics of Operations Research*, pp. 617–628.

Roth, A., and M. Sotomayor (1992): *Two-sided matching: A study in game-theoretic modeling and analysis*, vol. 18. Cambridge Univ Pr.

Shimer, R., and L. Smith (2000): "Assortative matching and search," *Econometrica*, 68(2), 343–369.

Su, X., and S. Zenios (2004): "Patient choice in kidney allocation: The role of the queueing discipline," *Manufacturing Service Operations Management*, 6(4), 280–301.

Su, X., and S. Zenios (2005): "Patient choice in kidney allocation: A sequential stochastic assignment model," *Operations research*, pp. 443–455.

——— (2006): "Recipient choice can address the efficiency-equity trade-off in kidney transplantation: A mechanism design model," *Management science*, 52(11), 1647.

Talreja, R., and W. Whitt (2008): "Fluid Models for Overloaded Multiclass Many-Server Queueing Systems with First-Come, First-Served Routing," *Management Science*, 54(8), 1513–1527.

Ünver, M. (2010): "Dynamic kidney exchange," *Review of Economic Studies*, 77(1), 372–414.

Wilson, R. (1987): *Game-Theoretic Analyses of Trading Processes*chap. 2, pp. 33–70. Advances in Economic Theory: Fifth World Congress.

Zenios, S. (1999): "Modeling the transplant waiting list: A queueing model with reneging," *Queueing systems*, 31(3), 239–251.

Zenios, S., G. Chertow, and L. Wein (2000): "Dynamic allocation of kidneys to candidates on the transplant waiting list," *Operations Research*, pp. 549–569.

Zou, J., S. Gujar, and D. Parkes (2010): "Tolerable manipulability in dynamic assignment without money," in *24th AAAI Conference on Artificial Intelligence (AAAI-10)*.

A FIFO with discounting

If $\delta < 1$ we have that:

$$\begin{aligned} U_\alpha(A^{(k)}) &= \sum_{t=k}^{\infty}(P_A)^k(1-P_A)^{t-k}\binom{t-1}{k-1}U_\alpha(A|t) = \\ &= \left(\frac{\delta P_A}{1-\delta\,(1-P_A)}\right)^k \times 1 - \frac{c}{1-\delta}\left(1-\left(\frac{\delta P_A}{1-\delta\,(1-P_A)}\right)^k\right) \end{aligned}$$

where $A^{(k)}$ denotes the k-th item that will arrive in the future, and $(P_A)^k(1-P_A)^{t-k}\binom{t-1}{k-1}$ is the probability that the arrival of the k-th item will be in t periods. An α agent will be willing to decline a B item if

$$U_\alpha(A^{(k)}) \geq U_\alpha(b)$$

which is true when

$$k \leq \frac{\log\left(\frac{c+v(1-\delta)}{c+1-\delta}\right)}{\log\left(\frac{\delta p_A}{1-\delta(1-p_A)}\right)}$$

B Omitted Proofs

Proof. (Claim 2) Consider an arbitrary allocation μ under some exogenous arrival process $\chi : \mathcal{A} \to \{t \geq 0\}$ which specifies the arrival time of each agent a. Assume the world ends after period T and denote by $\mathcal{A}_T = \{a \in \mathcal{A} | \chi(a) \leq T\}$ the set of agents that arrived before period T, and by $\mathcal{A}(\mu) = \mu(\{t_0 \leq t \leq T\})$ the set of agents that were assigned under μ before time T. The sum of agents utilities from 0 to T under μ is

$$WF_{0,T} = \sum_{t=0}^{T}\left((1-\xi_t)\cdot 1 + \xi_t \cdot v - c\cdot(t-\chi(\mu(t)))\right) - \sum_{a\in\mathcal{A}_T\setminus\mathcal{A}(\mu)} c\cdot(T-\chi(a))$$

where the first summation is over the utilities of agents that get assigned under μ and the second summation is over the utilities of agents of unassigned agents. Define the misallocation indicator ξ_t to take a value 1 if the agent $a = \mu(t)$ got a mismatched item and $\xi_t = 0$ if $a = \mu(t)$ got a preferred item. Rewriting, we get that

$$WF_{0,T} = \sum_{t=0}^{T} ((1 - \xi_t) \cdot 1 + \xi_t \cdot v) + \sum_{t=0}^{T} c \cdot (T - t) - \sum_{a \in \mathcal{A}_T} c \cdot (T - \chi(a))$$

Since the last two arguments do not depend on μ they will cancel out when we take difference between the welfare under two allocations μ, μ'. Therefore the relative welfare of an assignment depends only on the matching between item kinds and agent types. □

Proof. *(Proposition 6)* Let $\mathcal{T} : s_t \times \omega_t \mapsto a \times s_{t+1} \times \omega_{t+1}$ be the transition function of the waiting list policy, as defined in section 2. ω_t describes future item arrivals and unapproached agents, which are on the offer queue. As noted above in section 2 the distribution of ω_t is independent of the history. The dependence on information revealed in previous rounds is fully captured by the state of the assignment queue, as all agents that were previously approached by the mechanism are are yet unassigned are in the assignment queues.

When there is a positive number of agents in the A assignment queue only B items get offered to agents in the offer queue. When a B item is offered to a new agent that agent never joins the B assignment queue. Therefore no agents join the B assignment queue while the A assignment queue is not empty, and vice versa. Therefore the state space S includes only configurations of assignment queues where one of the queues is empty. By the agent's decision we know that the size of each assignment queue is bounded. Thus the state space of the mechanism is equivalent to $S = \{-K_\beta, \ldots, -1, 0, 1, 2, \ldots, K_\alpha\}$, where $s > 0$ indicates the number of agents in the A assignment queue and that the B

assignment queue is empty and similarly for $s < 0$.

The distribution of ω_t yields the Markovian transition function P . Suppose there are $k > 0$ α agents in the A assignment queue at the beginning of period t. With probability p_A an A item arrives. The A item is assigned to the first agent in the A assignment queue and the period ends with $k-1$ agents in the A assignment queue. With probability p_B a B item arrives. The B item is offered to the new agent who is first in the offer queue. With probability p_α the new agent will be an α and join the A assignment queue. The system will continue offering to agents until either a new agent will be a β agent, or until the A assignment queue has K_α agents and the next agent will take the B regardless of her type. Thus the probability of adding $l-k$ agents to the A assignment queue is $p_\alpha^{l-k}p_\beta$ when the assignment queue did not fill up, and $p_\alpha^{l-k}(p_\beta + p_\alpha) = p_\alpha^{l-k}$ when the queue fills up. The rest of the transition probabilities are similarly derived.

Proof. (Lemma 7) It is clear that all states in the Markov chain on $\hat{S}$ are recurrent. Let us denote the stationary distribution by π, where $\pi(k)$ is the stationary probability of state (k, ϕ) and $\pi^B(k)$ is the stationary probability of (k, B) (and likewise for π^A). The balance equations for $k > 0$ are:

$$\begin{aligned} \pi(k) &= p_A\pi(k+1) + p_\beta\pi^B(k) \\ \pi^B(k) &= p_\alpha\pi^B(k-1) + p_B\pi(k) \end{aligned}$$

The flow through the cut between $s \leq k$ and $s \geq k+1$ needs to be zero (see figure B.1), so we get the equation[23]

$$p_A\pi(k+1) = p_\alpha\pi^B(k)$$

[23]See Caldentey, Kaplan, and Weiss (2009).

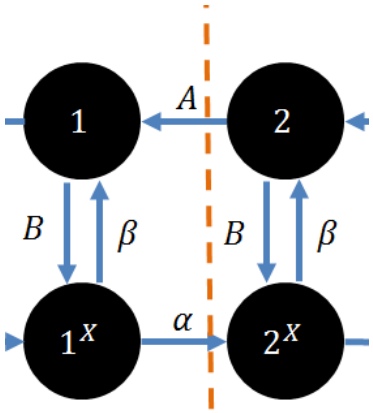


Figure B.1: The cut between $s \leq 1$ and $s \geq 2$

Together we get that for $k > 0$

$$\begin{aligned} \pi(k) &= p_A \pi(k+1) + p_\beta \frac{p_A}{p_\alpha} \pi(k+1) \\ &= p_A \frac{p_\alpha + p_\beta}{p_\alpha} \pi(k+1) \\ &= \frac{p_A}{p_\alpha} \pi(k+1) \end{aligned}$$

$$\begin{aligned} \pi^B(k) &= \frac{p_A}{p_\alpha} \pi(k+1) \\ &= \pi(k) \end{aligned}$$

from the balance equations for $(0, B)$ we get

$$\pi^B(0) = p_B \pi(0)$$

Therefore, for $k > 0$

$$\pi(k) = p_B \left(\frac{p_\alpha}{p_A} \right)^k \pi(0)$$

We get $\pi(0)$ by normalizing the total probability to 1. When $p_A = p_\alpha = p$ we have

$$
\begin{aligned}
1 &= \sum_{k=1}^{K_\alpha} \left(\pi(k) + \pi^B(k)\right) + \sum_{k=1}^{K_\beta} \left(\pi(-k) + \pi^A(-k)\right) + \pi(0) + \pi^B(0) + \pi^A(0) \\
&= 2 \sum_{k=1}^{K_\alpha} p_B \pi(0) + 2 \sum_{k=1}^{K_\beta} p_A \pi(0) + \pi(0) + p_B \pi(0) + p_A \pi(0) \\
&= 2 \pi(0) \left(p_B K_\alpha + p_A K_\beta + 1\right)
\end{aligned}
$$

implying that

$$
\pi(0) = \frac{1}{2} \frac{1}{(1-p)K_\alpha + pK_\beta + 1}
$$

When $p_A \neq p_\alpha$

$$
\begin{aligned}
1 &= \sum_{k=1}^{K_\alpha} \left(\pi(k) + \pi^x(k)\right) + \sum_{k=1}^{K_\beta} \left(\pi(-k) + \pi^x(-k)\right) + \pi(0) + \pi^B(0) + \pi^A(0) \\
&= 2 \sum_{k=0}^{K_\alpha} p_B \left(\frac{p_\alpha}{p_A}\right)^k \pi(0) + 2 \sum_{k=0}^{K_\beta} p_A \left(\frac{p_\beta}{p_B}\right)^k \pi(0) \\
&= 2\pi(0) \left(p_B \frac{p_\alpha \left(\frac{p_\alpha}{p_A}\right)^{K_\alpha} - p_A}{p_\alpha - p_A} + p_A \frac{p_\beta \left(\frac{p_\beta}{p_B}\right)^{K_\beta} - p_B}{p_\beta - p_B} \right) \\
&= 2\pi(0) \frac{p_A p_\beta \left(\frac{p_\beta}{p_B}\right)^{K_\beta} - p_B p_\alpha \left(\frac{p_\alpha}{p_A}\right)^{K_\alpha}}{p_A - p_\alpha}
\end{aligned}
$$

implying that

$$
\pi(0) = \frac{1}{2} \frac{p_A - p_\alpha}{p_A p_\beta \left(\frac{p_\beta}{p_B}\right)^{K_\beta} - p_B p_\alpha \left(\frac{p_\alpha}{p_A}\right)^{K_\alpha}}
$$

, which converges to the former expression when $p_\alpha \to p_A$. □

Proof. (Theorem 31) We build the scalable-optimal queueing policy by induction on $\bar{K}$, maximizing the set $\left\{v | K_\varphi(v) \geq \bar{K}\right\}$ at each stage. Without loss of generality we assume

that $\left\{v|K_\varphi(v) \geq \bar{K}\right\} = \{v|v \leq v_\varphi(K)\}$ where $v_\varphi(K) = \max\{v \mid \langle K, \varphi\rangle \text{ is Bf-IC for } v\}$, since a policy $\langle K, \varphi\rangle$ that is Bf-IC for v is also Bf-IC for any $v' \leq v$.[24] Therefore, the scalable optimal policy is given by the φ that maximizes $\{v_\varphi(K)\}_{K=1}^{\infty}$ under the lexicographic ordering. Since $v_\varphi(K)$ depends only on $\{\varphi(k,i)\}_{1\leq i\leq k\leq K}$ we derive φ inductively row by row, optimizing $\varphi(K,\cdot)$ given $\{\varphi(k,i)\}_{1\leq i\leq k\leq K-1}$.

Under all assignment probabilities φ we have $\varphi(1,1) = 1$ and $v_\varphi(1) = 1 - \frac{c}{p}$. To get the scalable-optimal policy we need to set $\varphi(2,\cdot)$ to maximize $v_\varphi(2)$. By lemma 25 we can write $v_\varphi(2) = \inf_\sigma \min_{k,i} g^B_\sigma(k,i)$. Holding σ fixed, let us define $\varphi(2,1) = x$, $\varphi(2,2) = 1-x$ and search for $0 \leq x^* \leq 1$ that maximizes the minimal $g^B_\sigma(k,i)$. Since $g^B_\sigma(2,1)$ is increasing in x and $g^B_\sigma(2,2)$ is decreasing in x the minimum of the two will be maximized by setting $x^* = \frac{1}{2}$ and $g^B_\sigma(2,1) = g^B_\sigma(2,2)$. Note that $x^* = \frac{1}{2}$ independently of σ and setting $x = \frac{1}{2}$ also gives $g^B_\sigma(1,1) \leq g^B_\sigma(2,1)$. Since $v_\varphi(2) \leq \inf_\sigma \min_i g^B_\sigma(2,i)$ we have thus showed that $\varphi(2,1) = \varphi(2,2) = \frac{1}{2}$ yields the maximal $v_\varphi(2)$.

We now continue by induction on K. Assume that $\varphi(k,i) = \frac{1}{k}$ for all $i \leq k \leq K-1$, and we need to show that setting $\varphi(K,i) = \frac{1}{K}$ maximizes $v_\varphi(K)$. Fix σ and denote $\varphi(K,K) = x$. Since $\varphi(K-1,i) = \varphi(K-1,j)$ for all $i,j \leq K-1$ we have that $\varphi(K,j) = \frac{1-x}{K-1}$. As we did for $K = 2$, we find the x^* that maximizes $\inf_\sigma \min_i g^B_\sigma(K,i)$ and show that it is optimal.

Denote by $w(k,i)$ the expected waiting time the agent in position (K,K) at the beginning of a period (before an item arrives). The expected wait $w(K,q+1)$ is given by:

$$w(K,K) = p_B \cdot (w(K,K)+1) + p_A \cdot (x \times 0 + (1-x) \times (w(K-1,K-1)+1))$$

From position $(K-1,K-1)$ the agent can only reach positions (k,i) with $i \leq K-1$, since agents only move positions ahead in the queue. Once in position $(K-1,K-1)$

[24] Wlog the set $\{v \mid \langle K,\varphi\rangle \text{ is Bf-IC for } v\}$ is closed since we defined Bf-IC using weak inequalities.

the agent will visit positions (k,i) for $i \leq k < K-1$ until she either gets assigned an item, or she returns to a some position (K,i). Let $\eta > 0$ denote the expected time from when the agent reaches position $(K-1,K-1)$ until she either gets assigned or moves to some position (K,i). Let $\rho < 1$ be the probability that the agent will reach some position (K,i) before getting assigned. Notice that η, ρ are independent of $\{\varphi(K,i)\}_{i=1..K}$. By symmetry, $w(k,i) = w(k,j)$ for all $i,j \leq K-1$. Therefore we can write :

$$w(K-1,K-1) = \eta + \rho\left(w(K,K-1)+1\right)$$

Finally, by the dynamic equations:

$$\begin{aligned} w(K,K-1) &= p_B(w(K,K-1)+1) + \\ &\quad + p_A\left(\frac{1-x}{K-1} \times 0 + \left(1 - \frac{1-x}{K-1}\right) \times \left(w(K-1,K-1)+1\right)\right. \end{aligned}$$

where we again used that by symmetry $w(K-1,K-1) = w(K-1,K-2)$.

Solving these equations and taking derivatives with respect to x gives

$$\frac{d}{dx}w(K,K) = -\frac{(K-1)^2(1-\rho)(p_A\eta+\rho)}{p_A\left((K-1)(1-\rho)+\rho(1-x)\right)^2} < 0$$

and

$$\frac{d}{dx}w(K,K-1) = \frac{(K-1)(p_A\eta+\rho)}{p_A\left((K-1)(1-\rho)+\rho(1-x)\right)^2} > 0$$

implying that $g^B_\sigma(K,K)$ is increasing with x (waiting time decreases with x) and $g^B_\sigma(K,i)$ is decreasing with x for $i \leq K-1$. By choosing $x = \frac{1}{K}$ we get that $g^B_\sigma(K,i) = g^B_\sigma(K,j)$ for all $i,j \leq K$, which gives us the unique maximum of $\min_{i\leq K} g^B_\sigma(K,i)$. Since under $\varphi(K,i) = \frac{1}{K}$ we also have that $g^B_\sigma(k,i) \geq g^B_\sigma(K,i)$ for all $i \leq k \leq K$, it follows that $\varphi(K,i) = \frac{1}{K}$ maximizes $\min_{i,k} g^B_\sigma(k,i)$. Since this was true for any σ we have that $\varphi(K,i) = \frac{1}{K}$ is the unique maximizer of $v_\varphi(K) = \inf_\sigma \min_{i,k} g^B_\sigma(k,i)$.

This shows that $SIRO$ produces the lexicographical maximum $\{v_\varphi(K)\}_{K=1}^{\infty}$ and is therefore the scalable optimal policy. $\square$