Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

A Nonasymptotic Approach to Analyzing Kidney Exchange Graphs

Yichuan Ding, Dongdong Ge, Simai He, Christopher Thomas Ryan

To cite this article:
Yichuan Ding, Dongdong Ge, Simai He, Christopher Thomas Ryan (2018) A Nonasymptotic Approach to Analyzing Kidney Exchange Graphs. Operations Research

Published online in Articles in Advance 24 Jul 2018

. https://doi.org/10.1287/opre.2017.1717

http://pubsonline.informs.org/journal/opre/

OPERATIONS RESEARCH
Articles in Advance, pp. 1–18
ISSN 0030-364X (print), ISSN 1526-5463 (online)

A Nonasymptotic Approach to Analyzing Kidney Exchange Graphs

Yichuan Ding,[a] Dongdong Ge,[b] Simai He,[b] Christopher Thomas Ryan[c]

[a] Sauder School of Business, University of British Columbia, Vancouver, British Columbia V6T 1Z2, Canada; [b] School of Information Management and Engineering, Shanghai University of Finance and Economics, 200083 Shanghai, China; [c] University of Chicago Booth School of Business, Chicago, Illinois 60637

Contact: daniel.ding@sauder.ubc.ca, http://orcid.org/0000-0003-3014-8973 (YD); ge.dongdong@mail.shufe.edu.cn (DG); simaihe@mail.shufe.edu.cn (SH); chris.ryan@chicagobooth.edu, http://orcid.org/0000-0002-1957-2303 (CTR)

Received: February 22, 2015
Revised: September 25, 2016; September 19, 2017
Accepted: November 15, 2017
Published Online in Articles in Advance: July 24, 2018

Subject Classifications: programming: integer: algorithms: branch-and-bound; probability: random walk; network/graphs: matchings
Area of Review: Games, Information, and Networks

https://doi.org/10.1287/opre.2017.1717

Abstract. We propose a novel methodology to study kidney exchange. Using a random graph model of kidney exchange, we propose a nonasymptotic approach to quantifying the effectiveness of transplant chains in reducing the number of unmatched highly sensitized patients. Our approach is based on a two-phase random walk procedure where random walks are used to allocate chains, followed by allocation in cycles. The benefit of random walks is that they preserve the probabilistic structure of residual graphs, greatly facilitating analysis. Our approach allows us to analytically show the benefit of chains, as opposed to transplantation in cycles only, in nonasymptotic (medium-sized) graphs. We also derive useful analytical bounds that illustrate the performance of our proposed allocation procedure and more general kidney allocation procedures. Our results complement previous findings from analytical results in large (limit) graphs and empirical results based on data from fielded kidney exchanges demonstrating the benefits of chains. Moreover, our analysis sheds light on the relative importance of chains versus cycles in kidney allocation. In particular, our results show prioritizing chains over easy-to-transplant cycles, as opposed to prioritizing those cycles over chains, improves performance and provides analytical bounds on the associated benefits. A detailed simulation study numerically verifies our main results and provides additional insights.

Funding: The research of the first author was partially supported by National Sciences and Engineering Research Council of Canada [NSERC Grant 436156-13]. The second author was partially supported by the Natural Science Foundation of China [NSFC Grant 11471205] and the Program for Innovative Research Team of Shanghai University of Finance and Economics. The third author was partially supported by the Natural Science Foundation of China [NSFC Grant 71771141 and Grant 71440014], the Program for Innovative Research Team of Shanghai University of Finance and Economics, and the Program for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning. The fourth author thanks the Booth School of Business for its generous research support.
Supplemental Material: The online appendix is available at https://doi.org/10.1287/opre.2017.1717.

Keywords: kidney exchange • random graph • long chains • pure death process • integer programming

1. Introduction

Allocating donated kidneys to deserving patients with end-stage renal disease is an important challenge in today's healthcare system. Kidney exchanges are an integral part of the allocation system and play a central role in live donation. Exchanges consist of pools of patients, each paired with a loved one willing to donate. A pair may be incompatible because of differences in blood type or other tissue sensitivities. The goal of the exchange is to swap donors among incompatible pairs to allow for more transplants. Exchanging kidneys among incompatible patient-donor pairs creates *cycles* of donation within an exchange.

More possibilities for exchange occur in the presence of *altruistic donors*—individuals who are willing to donate their kidney to any patient in need. That is, the kidney of an altruistic donor is not directed to any particular patient. For this reason, altruistic donors are also called nondirected donors (NDDs). There are alternative uses for the donated kidney of an NDD. Until relatively recently, the NDD kidneys were offered to the deceased donor wait list managed nationally under the aegis of the United Network of Organ Sharing (UNOS). However, within kidney exchanges, altruistic donors can initiate donation "chains." A *chain* starts with an altruistic donor offering a kidney to a compatible recipient. The paired donor of that recipient further donates his or her kidney to another compatible recipient, and so on. Since NDDs are not directed toward a particular recipient, a chain need not "cycle" back.

Allocating NDD kidneys among their alternative uses has sparked ethical and practical debate, including whether chains are needed at all (Roth et al. 2007,

Gentry et al. 2009, Ünver 2010, Woodle et al. 2010, Ashlagi et al. 2011). Theoretical results show that, under certain structures, short cycles are sufficient, eliminating the need for chains (Roth et al. 2007, Ünver 2010). On the other hand, empirical results and simulations consistently show that chains are important in practice. Ashlagi et al. (2012) and Dickerson et al. (2012b) resolve this discrepancy between theory and practice. Their analytical results reveal that the underlying sparseness of connections between patients and donors in the exchange is the main driver of the need for chains.

As the above demonstrates, analytical models of kidney exchange have been helpful in resolving debates and providing explanations of experimental data. As new empirical findings and practical issues have arisen, models have been adjusted to meet these challenges. The standard bearer of analytical work has been random graph models. These models approximate exchanges by generating, in a probabilistic fashion, nodes and arcs that represent recipient-donor pairs and compatibilities, respectively. Recent papers use asymptotic analysis as their primary theoretical workhorse; that is, they examine kidney exchange graphs as the number of nodes tends to infinity. The development of Ashlagi et al. (2012) described above is a primary example. Earlier theoretical findings (for instance, Roth et al. 2007) were based on *dense* random graphs and were inadequate to explain empirical findings. By revising the standard model to include *sparse* random graphs, Ashlagi et al. (2012) are able to theoretically justify the observed need for chains in fielded exchanges using asymptotic analysis. Asymptotic analysis has also been used to derive insights into incentives issues (Toulis and Parkes 2011), the effect of "failed" chains and cycles (Dickerson et al. 2013), myopic versus forward-looking considerations in dynamically allocating kidneys (Dickerson et al. 2012a, Ashlagi et al. 2013), and fairness issues (Dickerson et al. 2014).

However, asymptotic analysis has its limitations. Asymptotic results are best interpreted in the setting of "large" exchanges with many recipient-donor pairs, something not usually observed in practice (Melcher et al. 2012). Researchers in the area of kidney exchange are well aware of this limitation. Indeed, Ashlagi et al. (2012) state that analysis in "medium"-sized graphs should, in fact, be the target for analysis.

The main technical contribution of this paper is to develop a nonasymptotic methodology that applies to medium-sized exchanges. The core novelty of our methodology is to employ a random walk procedure with two distinct phases. The first phase is to allocate kidneys in chains via a memoryless random walk. After chains are removed, the second phase is to allocate via cycles.

As a tool for analysis, our two-phase procedure has many strengths, as evidenced by our analytical results in Sections 3 and 4. For example, we provide exact formulas and simple nonasymptotic analytical bounds for the tail probabilities and expectation of the random number of unmatched nodes after the termination of the first stage. Although nonasymptotic, these bounds can be used to recover asymptotic results (as demonstrated in Proposition 1). These bounds serve as inputs to further bound the expected number of unmatched nodes after both phases are implemented, assuming particular algorithms for assigning cycles in the second stage. These latter bounds allow us to assess the performance of our two-phase procedure and quantify the benefits of chains in medium-sized graphs.

More qualitatively, one of the challenges in both analyzing and managing kidney exchange graphs is trading off the benefits of chains versus cycles. From an analytical perspective, there are cases where chains are not needed (if the graph is sufficiently dense) and cases where chains are needed (if the graph is sufficiently sparse), as discussed above. However, there is little direct work on the relative importance of the *sequence* in which chains and cycles are removed. In particular, there remains open question of how to *prioritize* chains and cycles. As noted in the literature (see, for instance, Toulis and Parkes 2011), under the current system, hospitals may have an incentive to transplant short cycles locally and not submit these cases to an exchange. In this situation, easy-to-transplant cycles are de facto prioritized over chains, as many potential kidney-donor pairs cannot participate in long chains through an exchange. Our analysis sheds light on this issue and provides some analytical insight into the cost of this practice. In particular, we show via simulation and through analytical bounds that prioritizing chains over cycles outperforms prioritizing cycles over chains. This provides estimates of the cost of the practice of hospitals myopically transplanting cycles. Interestingly, we show through numerical simulations that the magnitude of this cost depends on the type of cycle prioritized. When only easy-to-transplant cycles are prioritized, the loss is much greater than when a wider class of cycles is prioritized.

Although our proposed procedure does not assign chains optimally (it uses random walk), there are a variety of cases where there is little loss overall. In kidney exchange, an optimal packing of chains may not be needed because of the presence of cycles. In our approach, the residual graph (the graph that remains after removing chains) maintains its initial probabilistic *density*, unaffected by random walk realizations in the first stage. Since our procedure assigns nodes to chains randomly, it does not target high-degree nodes that would allow for longer chains at the cost of increased sparsity at the cycle formation stage. Without

introducing additional sparsity, cycles are then sufficient to match many of the remaining patient-donor nodes. The dense structure of the residual graph allows probabilistic comparison between the original graph and residual graph. This idea is central to our analysis in Section 4.

In existing integer linear programming (ILP)-based implementations (see, for instance, Dickerson et al. 2014), the assignment of chains is computationally quite costly. For a given input graph, the number of cycles of bounded length is polynomial in the input size, whereas the number of chains grows exponentially in the input size when not capped. For the column-generation algorithms adapted in Dickerson et al. (2014), this can present computational challenges. The fact that random assignment of chains performs reasonably well (as evidenced by our simulations and confirmed by personal communication with John Dickerson 2014) may greatly reduce computational burden. We adapt this idea to construct a hybrid ILP implementation that uses both ILP and random walks, where random walks are used to generate a moderate number of candidate chains in the graph, and test its performance in Section 6. This hybrid algorithm has the benefit of being able to compute reasonable transplant plans faster, enabling consideration of more scenarios. Thus, we believe there is both theoretical and practical interest for using the concept of random walks in kidney exchange.

We organize the rest of this paper as follows. In Section 2 we introduce our random graph model of kidney exchange and propose our two-phase random walk procedure. Section 3 provides analysis of the nature of the graph at the termination of the first stage. Section 4 provides results on the nature of the graph after the termination of the second stage. Section 5 analyzes the impact of prioritizing cycles over chains (and vice versa). Section 6 contains our numerical experiments. Section 7 concludes and points to future work.

2. Analytical Framework

2.1. Random Graph Model

We consider the random graph model of kidney exchange proposed in Ashlagi et al. (2012). Similar models are employed in Ashlagi et al. (2013) and Dickerson et al. (2013). Careful justification of this model can be found in those papers.

The kidney exchange pool is modeled as a directed graph D, which contains two types of nodes: patient-donor nodes and NDD nodes. A directed arc (u,v) connects nodes u and v if the patient of node v is compatible with the donor of node u. Following Ashlagi et al. (2012), we suppress the issue of blood-type matchings and focus instead on tissue-type matching of donors and patients. Arc (u,v) appears in the graph with a probability that depends only on the tissue-type characteristics of node v. Furthermore, patient-donor nodes are classified into two categories: *high-sensitization nodes* and *low-sensitization nodes*. For brevity, we call high-sensitization nodes H-nodes and low-sensitization nodes L-nodes. Arc (u,v) appears in the graph with probability p_H (p_L) if v is an H-node (L-node), where u is an arbitrary node (not equal to v) in the graph. Throughout, we assume $p_H < 0.1 < p_L$. The assumption of two categories of patient-donor nodes is justified by empirical investigations found in Ashlagi et al. (2013), where it is shown that the probability a patient is compatible with a randomly selected donor follows a bimodal distribution.

Our model considers the possibility that NDDs and bridge donors (donors freed to donate to extend the length of a chain) renege before the time of transplantation. This is captured by the probability r. That is, every time a chain is extended, there is a probability r it terminates before the next link in the chain is transplanted.

We use the notation $D(h,\ell,t;p_H,p_L,r)$ to represent an exchange pool with h high-sensitization nodes, ℓ low-sensitization nodes, and t NDD donors, along with compatibility probabilities p_H and p_L and renege probability r. The proportion $\ell/(\ell+h)$ of low-sensitization nodes in the graph is denoted by λ. When certain parameters are understood as given, we drop them in our notation. For instance, when the focus is on the size of the exchange with probabilities fixed, we will write $D(h,\ell,t)$ instead of $D(h,\ell,t;p_H,p_L,r)$.

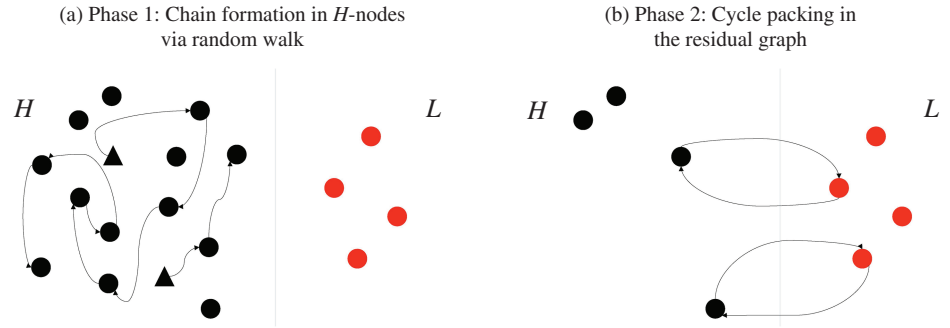
A *clearing* of the kidney exchange graph is a collection of disjoint cycles and chains that represent the patients and donors involved in transplantation. Cycles and chains must be disjoint since each patient can receive at most one kidney and every donor can give at most one kidney. A patient-donor node in a clearing is said to be *matched*, since the patient receives a kidney and the donor donates its kidney. In practice, kidney exchanges clear at regular intervals (weekly, monthly, or bimonthly) to balance the objectives of efficiency and fairness (see Dickerson et al. 2012b and Melcher et al. 2012 for details). Our model is static and considers only a single decision period.

2.2. Two-Phase Random Walk Procedure

We propose the following two-phase clearing procedure for $D(h,\ell,t;p_H,p_L,r)$ (hereafter simply called the *two-phase procedure*), illustrated in Figure 1.

***Phase* 1.** While there exists at least one NDD, initiate a chain starting from an NDD. At each step, grow the chain by adding an H-node accessible from the last node of the chain (referred to as a tail node). If there is more than one accessible H-node, randomly select one among them with equal probability. If no H-nodes are accessible, terminate the chain and remove all selected nodes in the chain (including the initiating

Figure 1. (Color online) Our Two-Phase Procedure



Note. Black disks are H-nodes, triangles are NDDs, and red disks are L-nodes.

NDD donor). Repeat until either all H-nodes have been removed or all NDDs have been consumed. Go to Phase 2.

Phase 2. Apply a cycle-packing algorithm on the residual graph that remains at the termination of Phase 1.

A few remarks on the procedure are in order. First, chains in Phase 1 are executed within the subgraph of H-nodes and NDD-nodes. There are no L-nodes in the chains of our procedure. Second, Phase 2 does not specify a cycle-packing algorithm. Our analysis in Section 4 provides theoretical bounds for the case where Phase 2 consists of bipartite matching between H- and L-nodes. Section 6 gives numerical results for when Phase 2 employs both two- and three-way cycles. Section 5 compares the performance of this algorithm to one where these phases are inverted.

The analytical power of the two-phase procedure comes from the fact we are able to derive upper bounds on the expected number of unmatched H-nodes after the termination of the algorithm and compare this to the expected number of unmatched H-nodes when only cycles are permitted. There are several steps to this analysis. In Section 3 we analyze Phase 1, focusing on probabilistic statements about how many H-nodes have been transplanted. Section 4 explores what happens after Phase 2, leveraging results from Phase 1.

3. Analysis of Phase 1

In this section, we define a two-dimensional-state stochastic process that tracks the progress of random walks, in terms of transplanting H-nodes and consuming NDD donors. Counting arguments yield exact probabilities associated with the random number of H-nodes left unmatched at the end of Phase 1. To yield more useful noncombinatorial bounds used in Section 4, we later define a potential function and construct martingales to get useful analytical estimates of the expected number of residual unmatched H-nodes.

Let $X(n)$ denote the number of unmatched H-nodes at the time when n nodes (either H-nodes or NDD donor nodes) have been removed from the original graph $D(h,\ell,t)$. Let $t(n)$ denote the number of remaining NDDs plus the one being used in the current chain at the time when n nodes (either H-nodes or NDD donor nodes) have been removed. The stochastic process is $\{(X(n),t(n)) \mid n \geqslant 0\}$. Each increment of "time" n denotes the removal of a node from the graph. When either the donor reneges or there are no compatible donors, an NDD node is removed and $t(n)$ is decremented by 1. We call this a "failure." When a compatible match is found and a patient gets a transplant, then an H-node is removed and $X(n)$ is decremented by 1. We call this a "success." By definition, $X(0)=h$, and $t(0)=t$.

This process of node removal eventually terminates. There are two conditions for termination. The first is that all NDD donors have been consumed, corresponding to $t(n)=0$. The second is that all H-nodes have been transplanted, corresponding to $X(n)=0$.

Observe that $\{(X(n),t(n)) \mid n \geqslant 0\}$ is a two-dimensional pure death process with absorbing states $\{(X,t) \mid X=0 \text{ or } t=0\}$. At each nonabsorbing state, the transition probability is given by

$$(X(n+1),t(n+1)) = \begin{cases} (X(n)-1,t(n)) & \text{w.p. } 1-r_{X(n)}, \\ (X(n),t(n)-1) & \text{w.p. } r_{X(n)}, \end{cases}$$

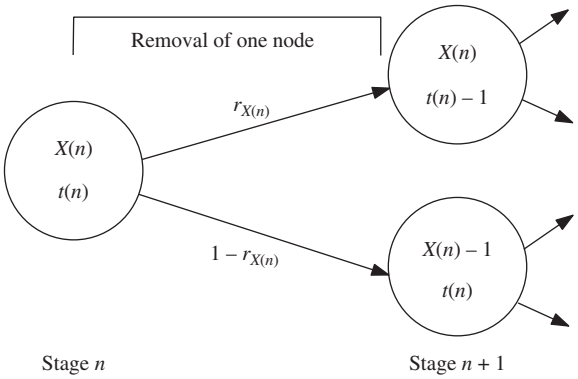
where

$$r_i = r + (1-r)(1-p_H)^i$$

gives the probability that either the tail node reneges or the tail node cannot find an accessible H-node. From this definition we see that $\{(X(n),t(n)) \mid n \geqslant 0\}$ is Markovian. Figure 2 provides a visual representation.

When the graph contains i H-nodes, the number of NDDs consumed to reduce the number of unmatched

Figure 2. The Stochastic Process $\{(X(n), t(n)) \mid n \geqslant 0\}$



H-nodes by 1 is a geometric random variable with success probability $1 - r_i$ and mean

$$\mu_i := \frac{r_i}{1 - r_i}. \tag{1}$$

For the ease of the subsequent analysis, we define the following potential function:

$$T(n) = \sum_{i=n+1}^{\bar{M}} \mu_i,$$

where $\bar{M}$ is a large constant integer. Given an integer $n \geqslant 0$, the function $T(n)$ calculates the expected number of NDDs needed to reduce the number of H-nodes from $\bar{M}$ to n. In the special case of $r = 0$, $\mu_i = (1 - p_H)^i / (1 - (1 - p_H)^i)$ and $\sum_{i=1}^{\infty} \mu_i < \infty$. In this case, we can safely assign $\bar{M} = +\infty$ without worrying that $T(n)$ diverges to infinity. For $r = 0$, the potential function is

$$T^0(n) := \sum_{i=n+1}^{\infty} \mu_i. \tag{2}$$

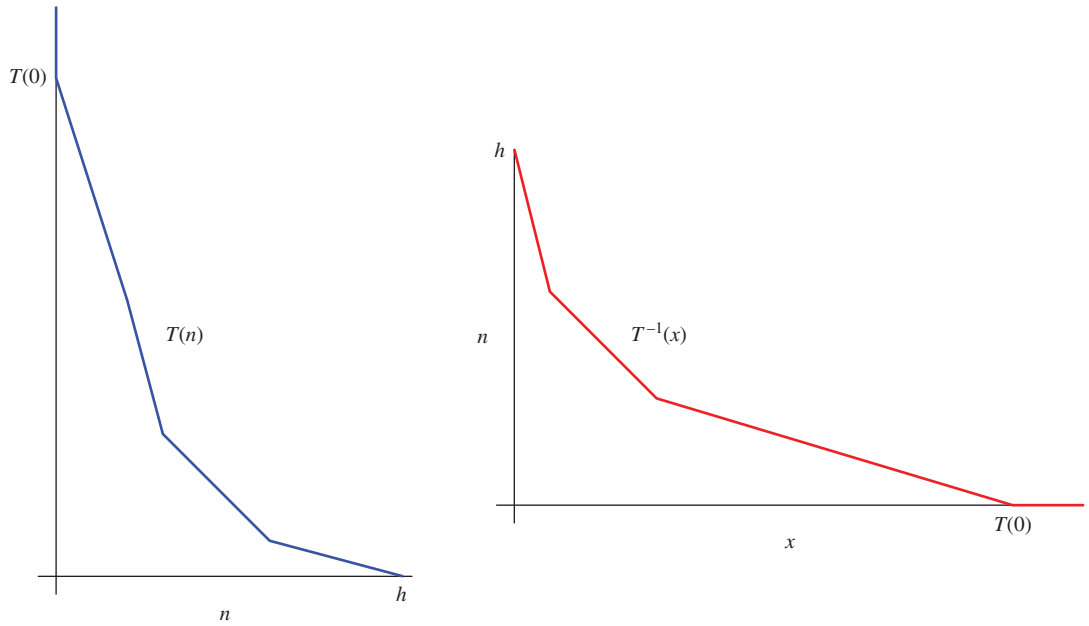
Observe that $T(n)$ is a strictly decreasing function on the discrete domain $0, 1, \ldots, \bar{M}$. We extend $T(\cdot)$ to be defined over the continuous domain $[0, \bar{M}]$ via piecewise linear interpolation. This makes the inverse function T^{-1} well defined on the range $[0, T(0)]$ of T where $T(0) < \infty$. For $x \geqslant T(0)$, we take $T^{-1}(x) = 0$, which will not modify the monotonicity of $T^{-1}(\cdot)$. Under this extension, both T and T^{-1} are convex functions because T has increasing differences: $T(i) - T(i-1) = -\mu_i$, and μ_i is decreasing in i since r_i is decreasing in i. See Figure 3 for a visualization.

We can define the random stopping time as the first time when either all H-nodes or all NDDs are matched; that is,

$$\tau_0 = \min\{n \mid t(n) = 0 \text{ or } X(n) = 0\}.$$

The time of termination of Phase 1 is precisely τ_0. We seek distributional information on the random number $X(\tau_0)$ of unmatched H-node patients at the time of termination of Phase 1. By the Markovian property of the process, when there are multiple NDDs, whether chains are selected simultaneously or sequentially does not affect the distribution of $X(\tau_0)$. That is, we may either grow multiple chains simultaneously or complete one chain and then start another, and we result in the same distribution of $X(\tau_0)$.

Figure 3. (Color online) The Functions T and T^{-1}



To make the dependence of $X(\tau_0)$ on h and t explicit, let $Y_{h,t}$ denote the value of $X(\tau_0)$ when the initial graph is $D(h,\ell,t)$. We are interested in the following performance metrics: (a) the tail probability $\Pr(Y_{h,t} \leqslant k)$ for a given nonnegative integer k and (b) the expectation of $Y_{h,t}$.

Theorem 1. *For a random kidney exchange graph $D(h,\ell,t)$,*

(a)

$$\Pr(Y_{h,t} \leqslant k) = \begin{cases} \prod_{i=k+1}^{h}(1-r_i) & \text{when } t=1, \\ \prod_{i=k+1}^{h}(1-r_i) \sum_{k \leqslant i_{t-1} \leqslant \cdots \leqslant i_1 \leqslant h} \prod_{j=1}^{t-1} \xi_k(i_j) & \\ \qquad \text{when } t \geqslant 2, \end{cases} \tag{3}$$

where $\xi_k(i) = r_i$ for $i > k$, and $\xi_k(i) = 1$ if $i \leqslant k$; and

(b) *consequently,*

$$\mathbb{E}[Y_{h,t}] = \begin{cases} \sum_{k=0}^{h-1}\left(1 - \prod_{i=k+1}^{h}(1-r_i)\right) & \text{when } t=1, \\ \sum_{k=0}^{h-1}\left(1 - \prod_{i=k+1}^{h}(1-r_i) \sum_{k \leqslant i_{t-1} \leqslant \cdots \leqslant i_1 \leqslant h} \prod_{j=1}^{t-1} \xi_k(i_j)\right) & \\ \qquad \text{when } t \geqslant 2. \end{cases}$$

Proof. (a) Note that $Y_{h,1}$ represents the number of unmatched H-nodes after matching with a single chain. The only way for $Y_{h,1} \leqslant k$ is for there to be a string of consecutive successes in extending the chain to reduce the number of unmatched H-nodes from h to k. By independence, this happens with probability $\prod_{i=k+1}^{h}(1-r_i)$.

The event $\{Y_{h,t} \leqslant k\}$ contains all scenarios where there are less than t failures in the course of removing $h-k$ H-nodes. Suppose there are $t' \leqslant t-1$ failures before $X(n)$ hits k. For $j = 1, 2, \ldots, t'$, we let i_j denote the number of H-nodes remaining in the graph at the time of the jth failure. The failure rate $\xi_k(i_j)$ at i_j is thus r_{i_j}. Whereas for $j = t'+1, \ldots, t-1$, we simply assign $i_j = k$ and $\xi_k(i_j) = 1$, as these failures happen after $X(n)$ hits k and therefore do not contribute to the event $\{Y_{h,t} \leqslant k\}$. Thus, we derive the tail probability for $Y_{h,t}$ as

$$\Pr(Y_{h,t} \leqslant k) = \prod_{i=k+1}^{h}(1-r_i) \sum_{k \leqslant i_{t-1} \leqslant \cdots \leqslant i_1 \leqslant h} \prod_{j=1}^{t-1} \xi_k(i_j).$$

(b) The expression for $\mathbb{E}(Y_{h,t})$ directly follows from the equation $\mathbb{E}[X] = \sum_{k=0}^{\infty} \Pr(X > k)$ for nonnegative discrete random variables and the fact that $\Pr(Y_{h,t} > h) = 0$. □

The above combinatorial expressions for the tail probabilities and expectation of $Y_{h,t}$ are precise but difficult to work with. The next result provides bounds that involve the potential function T and are more amenable to later analysis.

Theorem 2. *For a random kidney exchange graph $D(h,\ell,t)$,*

$$\Pr(Y_{h,t} \leqslant k) \geqslant \begin{cases} \exp(T(h)-T(k)) & \text{if } t=1, \\ \exp(T(h)-T(k)) \dfrac{(1+\sum_{i=k+1}^{h} r_i)^{t-1}}{(t-1)!} & \text{if } t \geqslant 2. \end{cases}$$

The proof of this theorem is in Online Appendix EC.1 and uses bounds on sums in terms of the exponential function. The result gives rise to a simpler corollary when $r = 0$ that is used in later results.

Corollary 1. *For a random kidney exchange graph $D(h,\ell,t)$ in the special case of $r = 0$, we have*

$$\Pr(Y_{h,t} \leqslant k) \geqslant \begin{cases} 1 - \dfrac{1}{p_H}\left((1-p_H)^{k+1} - (1-p_H)^{h+1}\right) & \text{when } t=1, \\ \exp\left(-\dfrac{(1-p_H)^{k+1}}{p_H(1-(1-p_H)^{k+1})}\right) \dfrac{(1+\sum_{i=k+1}^{h} r_i)^{t-1}}{(t-1)!} & \\ \qquad \text{when } t \geqslant 2. \end{cases} \tag{4}$$

Proof. We first prove inequality (4) for the $t \geqslant 2$ case. By Theorem 2, $\Pr(Y_{h,t} \leqslant k) \geqslant \exp(T(h) - T(k))(((1 + \sum_{i=k+1}^{h} r_i)^{t-1})/(t-1)!)$ for $t \geqslant 2$. Thus, to prove inequality (4), it suffices to show that $\exp(T(h) - T(k)) \geqslant \exp(-(1-p_H)^{k+1}/(p_H(1-(1-p_H)^{k+1})))$.

When $r = 0$, we have $r_i = (1-p_H)^i$, and therefore $\mu_i = (1-p_H)^i/(1-(1-p_H)^i)$ by (1). We can then lower bound $\exp(T(h) - T(k))$ as follows:

$$\begin{aligned} \exp(T(h)-T(k)) &= \exp\left(-\sum_{i=k+1}^{\bar{M}} \mu_i\right) \\ &= \exp\left(-\sum_{i=k+1}^{\bar{M}} \frac{(1-p_H)^i}{1-(1-p_H)^i}\right) \\ &\geqslant \exp\left(-\sum_{i=k+1}^{\infty} \frac{(1-p_H)^i}{1-(1-p_H)^{k+1}}\right) \\ &\geqslant \exp\left(-\frac{(1-p_H)^{k+1}}{p_H(1-(1-p_H)^{k+1})}\right). \end{aligned}$$

We have thus proved inequality (4).

For $t = 1$, we use backwards induction on k. When $k = h$, $\Pr(Y_{h,1} \leqslant h) = 1 \geqslant 1 - (1/p_H)((1-p_H)^{h+1} - (1-p_H)^{h+1})$. So we have proved the base case of $k = h$. We next show that inequality (4) for the $t = 1$ case holds for all $k > 0$, by assuming that it holds for $k+1$. We derive an upper bound for $\Pr(Y_{h,1} \leqslant k)$ as follows:

$$\begin{aligned} \Pr(Y_{h,1} \leqslant k) &= \Pr(Y_{h,1} \leqslant k+1) \Pr(Y_{h,1} \leqslant k \mid Y_{h,1} \leqslant k+1) \\ &= \left(1 - \frac{1}{p_H}\left((1-p_H)^{k+2} - (1-p_H)^{h+1}\right)\right)(1-(1-p_H)^{k+1}) \\ &\geqslant 1 - (1-p_H)^{k+1} - \frac{1}{p_H}(1-p_H)^{k+2} + \frac{1}{p_H}(1-p_H)^{h+1} \end{aligned}$$

$$= 1 - \frac{1}{p_H}(1-p_H)^{k+1} + \frac{1}{p_H}(1-p_H)^{h+1}$$
$$= 1 - \frac{1}{p_H}((1-p_H)^{k+1} - (1-p_H)^{h+1}),$$

where the first equality follows from Markov property, the second equality follows from the induction assumption, and the inequality holds by omitting the term $(1/p_H)((1-p_H)^{k+2} - (1-p_H)^{h+1})(1-p_H)^{k+1}$. This completes the induction. □

The above bounds are applicable to exchange graphs of arbitrary size. Although our focus is on nonasymptotic analysis, they can be leveraged in asymptotic settings to derive results similar to those in Ashlagi et al. (2012, 2013). The following result demonstrates this approach. Recall that h and ℓ denote the number of high- and low-sensitization nodes, respectively, and $\lambda = \ell/(h+\ell)$ denotes the proportion of low-sensitization nodes in the graph.

Proposition 1. *Suppose $r = 0$, h is in the order of $1/p_H^{1+\epsilon}$ for some $\epsilon > 0$, and both λ and p_L are fixed constants. If $t \geqslant 1$, then with probability approaching 1, the exchange graph has a perfect clearing (that is, all nodes are transplanted) as $p_H \to 0$.*

Proof. In (4), suppose $k = (c/p_H)\log(1/p_H)$ for $c > 1$. This yields $\Pr(Y_{h,1} \leqslant k) \geqslant \exp(-p_H^{c-1})$, which converges to 1 as $p_h \to 0$. Thus with probability approaching 1, no matter how large the original graph is, only one NDD is sufficient to reduce the number of H-nodes to the order of $O((1/p_H)\log(1/p_H))$. Since $t \geqslant 1$, with probability approaching 1, the number of H-nodes remaining unmatched after Phase 1, denoted by h', is in the order of $O((1/p_H)\log(1/p_H))$. We claim that when $p_H \to 0$, with probability approaching 1, all of those h' H-nodes can be matched to L-nodes using two-way cycles.

To prove this claim, we construct an undirected bipartite graph $\tilde{G} = (V_H \cup V_L, \tilde{E})$ with partitioned node sets $V_H := \{\text{all remaining } h' \text{ unmatched } H\text{-nodes}\}$ and $V_L := \{\text{all } L\text{-nodes}\}$, and an undirected edge set $\tilde{E} = \{(v_H, v_L) \mid v_H \in V_H, v_L \in V_L, (v_H, v_L), (v_L, v_H) \in E\}$. Each edge occurs with probability of $p_H p_L$—the probability of having a two-way cycle between an H-node and an L-node. According to the marriage theorem (Hall 1935), if the H-nodes cannot be matched in $\tilde{G}$, then there exists a "bad" pair of subsets $A \subset V_H$ and $B \subset V_L$ with $a = |A| > b = |B|$, and the set B contains all nodes adjacent to nodes in A. Without loss of generality, we may assume that (A, B) is a minimal bad pair, which means that there is no bad pair (A', B') with $A' \cup B' \subset A \cup B$. When (A, B) is a minimal bad pair, we must have $b = a - 1$. The probability that any nodes outside B are not linked to any node inside A is given by $(1 - p_H p_L)^{(\ell-b)a} = (1 - p_H p_L)^{(\ell-a+1)a}$. Since there are at most C_h^a and C_ℓ^b candidates for a minimal bad pair (A, B) of sizes a and b, respectively, the probability that at least one minimal bad pair exists of this size can be upper bounded by

$$\begin{aligned}
&\sum_{a=1}^{h'} C_{h'}^a C_\ell^{a-1} (1-p_H p_L)^{(\ell-a+1)a} \\
&\quad \leqslant \sum_{a=1}^{h'} \frac{(h'\ell)^a}{(a!)^2} (1-p_H p_L)^{(\ell-a+1)a} \\
&\quad \leqslant \sum_{a=1}^{h'} \frac{1}{a!} [\ell h' (1-p_H p_L)^{\ell-a+1}]^a \\
&\quad \leqslant \exp(\ell h' (1-p_H p_L)^{\ell-h'+1}) - 1 \\
&\quad \leqslant \exp(\ell h' \exp(-p_H p_L (l - h' + 1))) - 1. \qquad (5)
\end{aligned}$$

Note that $\ell = O(1/p_H^{1+\epsilon}) \gg h'$; thus $\ell - h' + 1 = O(1/p_H^{1+\epsilon})$ and $p_H p_L(\ell - h' + 1) = O(1/p_H^{\epsilon}) \geqslant 2\log(1/p_H)$ when p_H is sufficiently small. Therefore, the right-hand side of (5) is upper bounded by $\exp(\ell h' \exp(2\log(1/p_H))) - 1 = \exp(\ell h' p_H^2) - 1 = \exp(O(p_H^{1-\epsilon})) - 1 \to 0$, implying that the probability of the occurrence of a bad pair converges to 0 when $p_H \to 0$. Therefore, with probability approaching 1, all the H-nodes can be matched using H-L two-way cycles. After removal of all the H-nodes, the remaining subgraph contains L-nodes only. Each pair of L-nodes can be matched with a constant probability of p_L^2. Because the size of the remaining graph is $\ell - h' = O(1/p_H^{1+\epsilon}) \to \infty$ as $p_H \to \infty$, we know that a perfect matching exists by the well-known Erdős-Rényi theorem (Erdős and Rényi 1959). □

Remark 1. When h has an order of $1/p_H^{1+\epsilon}$, theorem 5.6(2) of Ashlagi et al. (2012) proves that all nodes can be matched using k-way cycles if $\lambda > 1/k$ or using chains with length $\leqslant m$ if $t \geqslant ((1-\lambda)/m)h$. Proposition 1 states that if we have a single chain of potentially infinite size (which is a weaker assumption than having $((1-\lambda)/m)h$ chains, each with length $\leqslant m$, with respect to the purpose of matching the $(1-\lambda)h$ H-nodes) and a fixed proportion of L-nodes, we can clear all nodes. This complements the result of Ashlagi et al. (2012).

We now turn to deriving noncombinatorial lower and upper bounds on $\mathbb{E}[Y_{h,t}]$ for later analysis. Of course, one could combine Theorems 1 and 2 to achieve this, but a different method will yield cleaner bounds. The proof of the following result (in particular, part (c)) is involved and can be found in Online Appendix EC.2. It uses the martingale theory and convexity of the potential function T and its inverse T^{-1}.

Theorem 3. *The following conditions hold*:

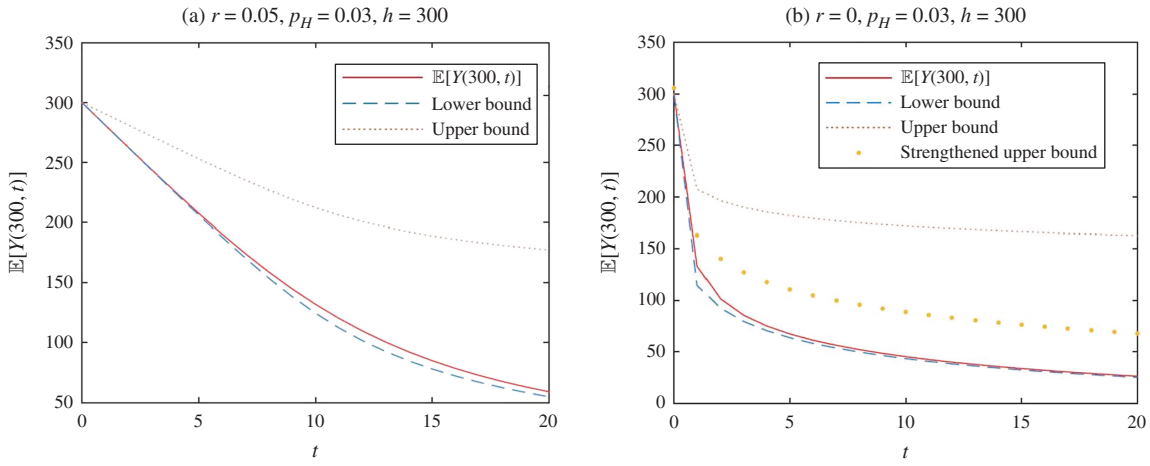
(a) *The sequence $\{T(X(n)) + t(n) \mid n \geqslant 0\}$ is a martingale. As a consequence,*

$$\mathbb{E}[Y_{h,t}] \geqslant T^{-1}(T(h) + t). \qquad (6)$$

(b) *The sequence $\{(X(n) + T^{-1}(T(X(n)) + t(n)))/2 \mid n \geqslant 0\}$ is a supermartingale. As a consequence,*

$$\mathbb{E}[Y_{h,t}] \leqslant \tfrac{1}{2}(T^{-1}(T(h) + t) + h). \qquad (7)$$

Figure 4. (Color online) The Expected Number of Unmatched H-Nodes $\mathbb{E}[Y(h,t)]$ and Its Lower Bound (6), Upper Bound (7), and the Strengthened Upper Bound (8)



Note. The strengthened upper bound is applicable only to the $r = 0$ case.

(c) *In the case of* $r = 0$ *and* $p_H \leqslant 0.1$, *we have the following strengthened upper bound*:

$$\mathbb{E}[Y_{h,t}] \leqslant \frac{1}{p_H} \log\left(1 + \frac{1}{(T^0(h) + \frac{1}{4}t)p_H}\right), \tag{8}$$

where $T^0(h)$ *defined in Equation* (2) *is the expected number of NDDs required to reduce the number of H-nodes from* $+\infty$ *to* h.

Figures 4(a) and (b) plot actual values of $\mathbb{E}[Y(h,t)]$ versus the bounds in Theorem 3 for $p_H = 0.03$, $h = 300$, and $r = 0.05$ (in (a)) and $r = 0$ (in (b)) (note the values of p_L and λ are not relevant because we are just working within the H-subgraph). The lower bound from (a) is quite tight. The upper bound from (b) is not so tight, but it nonetheless helps us to understand the asymptotic behavior of $\mathbb{E}[Y(h,t)]$. When $r = 0$, the bound in (c) is much tighter and implies that the number of unmatched H-nodes after Phase 1 is upper bounded by $O((1/p_H)\log(1/p_H))$. (Since $T^0(h) + \frac{1}{4}t$ is usually small, $1 + 1/((T^0(h) + \frac{1}{4}t)p_H) = O(1/p_H)$.)

4. Analysis of Phase 2

The goal of this section is to provide analytical bounds on the number of unmatched H-nodes that are left after termination of Phases 1 and 2. This analysis proceeds by comparison against a benchmark—namely, the number of unmatched H-nodes that remain if only Phase 2 was implemented from the beginning. In other words, we are interested in the *net* benefit of our procedure to reduce the number of unmatched H-nodes *beyond* what *could have* been transplanted via cycles alone. Our performance metric does not include the unmatched L-nodes, as it is easier to clear all the L-nodes even using cycles only. Moreover, keeping L-nodes for a later clearing may even be preferred (Ashlagi et al. 2013). As in the previous section, the underlying memoryless property of random walks and convexity arguments play a pivotal role here, as in the analysis of Phase 1.

A first challenge is to understand the random structure of the residual graph that remains at the termination of Phase 1. We show that this residual graph is again a graph of the form $D(h', \ell, t)$, where parameters ℓ, t, p_H, p_L, and r are fixed and $h' \leqslant h$. Throughout this section we suppose $h' > 0$ and all NDDs are consumed during Phase 1. The case where all H-nodes are matched before all NDDs are consumed is a somewhat uninteresting special case since it is very unlikely to occur in practice and so does not warrant further analysis.

Lemma 1. *Let* $D = D(h, \ell, t)$ *denote the initial random graph. Suppose during Phase* 1 *that* $h - h'$ *H-nodes are transplanted and removed, and let* $\mathcal{R}(h')$ *denote the conditional residual graph—that is, the random graph resulting from* D *by removing exactly* $h - h'$ *H-nodes in chains via Phase* 1. *Then the edge distribution in* $\mathcal{R}(h')$ *is identical to the random graph* $D' = D(h', \ell, 0)$.

Proof. Fix an ordering of the H-nodes in the starting graph D. There are $\binom{h-h'+t}{t}$ scenarios in which one starts with the graph D and ends with a residual graph that has h' remaining H-nodes and 0 NDD nodes. We simply need to distribute t failures among removing each of the $h - h'$ H-nodes. Because of the memoryless property of random walk, each of these residual graphs is isomorphic as a random graph to D'. Now, consider the conditional random graph $\mathcal{R}(h')$. Since each of the scenarios is disjoint, we can conclude that the edge distribution in the remaining graph $\mathcal{R}(h')$ is identical to the random graph D'. □

As a first step, consider a simple instantiation of Phase 2 that involves bipartite matching. This is admittedly not the optimal choice of algorithm for Phase 2, but it nonetheless forms a bedrock for our analysis. Bipartite matching algorithms are for undirected graphs, and so we construct an undirected version of the relevant part of the kidney exchange graph. The set of nodes is partitioned into H- and L-nodes, and we only consider matches between H- and L-nodes. An undirected edge uv is in this undirected version of the random bipartite graph $\hat{D}$ if and only if (u,v) and (v,u) are *both* directed edges in the original directed graph D. Hence, the probability undirected edge uv appears is $p_H p_L$. We let $\mathscr{C}_{HL}$ denote the (random) set of undirected edges of this bipartite graph. Phase 2 consists of using an algorithm to find maximal matching among the edges of $\mathscr{C}_{HL}$ (for instance, using linear programming). This algorithm generates a maximal cardinality matching $\mathscr{M}$, which is a subset of the (random) set $\mathscr{C}_{HL}$ (we also use this notation later in Section 5).

Let $f(h)$ denote the expected number of H-nodes remaining when nodes are matched using bipartite matching from the outset (that is, Phase 1 is not implemented) on a graph with h H-nodes. In the definition of f, the number of L-nodes and the probabilities p_H, p_L, and r are all fixed constants, and we write f simply as a function of h. Lemma 1 implies the following pivotal corollary.

Corollary 2. *The expected number of unmatched H-nodes after Phase 1 and Phase 2 is implemented on $D(h,\ell,t)$, given that Phase 1 eliminates $h-h'$ H-nodes, is $f(h')$.*

Thus, the value of implementing chains versus not implementing chains can be partially understood by comparing $\mathbb{E}[f(h')]$ to $f(h)$, where h' is now a random variable that represents the number of H-nodes remaining in the graph D after Phase 1 terminates; that is, $h' = Y_{h,t}$, where $Y_{h,t}$ is the number of unmatched H-nodes. The expectation in $\mathbb{E}[f(Y_{h,t})]$ is over the distribution of unmatched H-nodes after Phase 1. Whereas we do not have an explicit characterization of f (making a direct evaluation of $\mathbb{E}[f(Y_{h,t})]$ difficult), Section 3 does provide good estimates of $\mathbb{E}[Y_{h,t}]$. To leverage these estimates, we show (in Lemma 2) that the function f is convex. The proof of the lemma is in Online Appendix EC.3. It involves the linear programming formulation of bipartite matching and submodularity arguments.

Lemma 2. *The expected number $f(h)$ of unmatched H-nodes remaining after running the bipartite matching algorithm described above to $D(h,\ell,t)$ is convex in h.*

We leverage the convexity of f to bound the performance of the two-phase procedure when Phase 2 implements bipartite matching. This involves the following notation. Let

$$v_C^* := \mathbb{E}[f(Y_{h,t})] \tag{9}$$

denote the expected number of unmatched H-nodes after completion of both Phase 1 and the bipartite matching algorithm in Phase 2 on $D(h,\ell,t)$.

Theorem 4. *The following holds*:

$$\begin{aligned} v_C^* &\leqslant \frac{f(h)}{h}\mathbb{E}[Y_{h,t}] \\ &\leqslant \frac{1}{h}\left((h-\ell)^+ + \frac{(1-p_Lp_H)^{|h-\ell|+1} - (1-p_Lp_H)^{\max\{h,\ell\}+1}}{p_Lp_H}\right) \\ &\quad \cdot \mathbb{E}[Y_{h,t}]. \end{aligned} \tag{10}$$

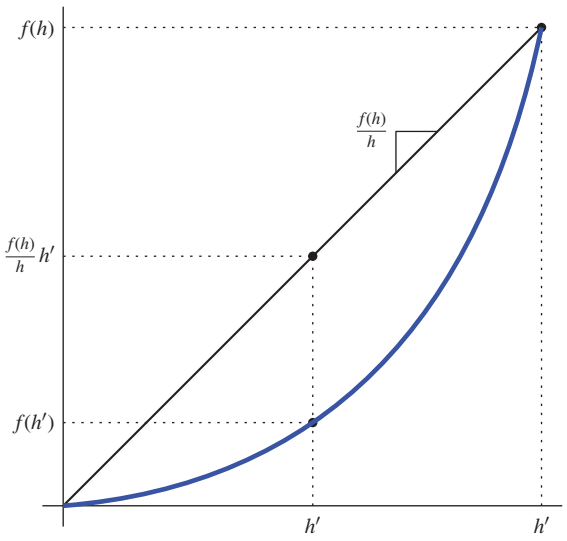
Proof. By Lemma 2, we know $f(h') \leqslant (f(h)/h)h'$ for any fixed $h' \leqslant h$, since f is convex. See Figure 5 and observe that $f(0) = 0$. Then, by the monotonicity of expectation, we have

$$\mathbb{E}[f(Y_{h,t})] \leqslant \mathbb{E}\left[\frac{f(h)}{h}Y_{h,t}\right] = \frac{f(h)}{h}\mathbb{E}[Y_{h,t}]. \tag{11}$$

This is the first inequality in (10).

The second inequality comes from bounding $f(h)$ from above. We derive the bound by analyzing the following naïve algorithm to match H-nodes. When $h > \ell$, sequence the L-nodes in an arbitrary order. For each L-node, attempt to match to an H-node using available edges (which correspond to two-way cycles in the directed graph). If successful, remove the matched pair and proceed to the next L-node. The algorithm terminates when all the L-nodes have been matched. According to this algorithm, there are at least $h-i+1$ unmatched H-nodes when the ith L-node is next to be matched. Thus, the probability of matching the ith L-node is at least $1-(1-p_Hp_L)^{h-i+1}$. Summing up these

Figure 5. (Color online) Leveraging the Convexity of f in the Proof of Theorem 4



probabilities, we derive a lower bound for the expected number of matched H and L nodes:

$$\sum_{i=1}^{\ell}(1-(1-p_Hp_L)^{h-i+1}) = \ell - \frac{(1-p_Hp_L)^{h-\ell+1}-(1-p_Hp_L)^{h+1}}{p_Hp_L}.$$

This leads to the upper bound on unmatched H-nodes:

$$f(h) \leqslant h-\ell+\frac{(1-p_Hp_L)^{h-\ell+1}-(1-p_Hp_L)^{h+1}}{p_Hp_L}. \quad (12)$$

When $h<\ell$, we propose a similar algorithm. This time, we use two-way cycles to match the H-nodes sequentially. When the algorithm tries to match the ith H-node, there are at least $\ell-i+1$ unmatched L-nodes remaining, so the probability of matching the ith H-node is at least $1-(1-p_Hp_L)^{\ell-i+1}$. Using a similar logic as above, we derive the upper bound:

$$f(h) \leqslant \frac{(1-p_Hp_L)^{\ell-h+1}-(1-p_Hp_L)^{\ell+1}}{p_Hp_L}. \quad (13)$$

Together, (12) and (13) imply the following upper bound on $f(h)$ that applies to both cases ($h>\ell$ and $h<\ell$):

$$f(h) \leqslant (h-\ell)^+ + \frac{(1-p_Lp_H)^{|h-\ell|+1}-(1-p_Lp_H)^{\max\{h,\ell\}+1}}{p_Lp_H}.$$

Plugging this upper bound into (11) yields the second inequality in (10). □

Of course, one may wonder how Phase 1 and Phase 2, as currently specified, compare in performance to more sophisticated clearing algorithms. This is a major topic in our numerical experiments in Section 6. For now, we show how to analytically bound the performance of an optimal two-way cycle-packing algorithm applied to the original graph in comparison to the performance using the two-phase procedure with bipartite matching in Phase 2. We let v_2^* denote the expected number of unmatched nodes when applying an optimal two-way cycle-packing algorithm to the original random graph $D(h,\ell,t)$.

Theorem 5. *Recalling $\lambda=\ell/(h+\ell)$, we have*

$$\frac{v_C^*}{v_2^*} \leqslant \frac{\mathbb{E}[Y_{h,t}]}{h}\frac{1-2\lambda}{1-2\lambda-(1-\lambda)p_H^2h}, \quad (14)$$

where v_C^* is defined in (9) and the denominator in the right-hand side is assumed to be positive.

Proof. Suppose when running the optimal matching algorithm using two-way cycles that the number of H-nodes that are matched by H-H cycles is n_1, and that of H-L cycles is n_2. The expected number of H-nodes matched in H-H cycles is upper bounded by the expected total number of H-H cycles in the H-subgraph. Hence, $\mathbb{E}[n_1] \leqslant h(h-1)p_H^2 \leqslant p_H^2h^2$. Clearly, $\mathbb{E}[n_2] \leqslant h-f(h)$. Therefore, $\mathbb{E}[n_1+n_2] \leqslant h-f(h)+p_H^2h^2$. Thus, $v_2^* \geqslant f(h)-p_H^2h^2$. By the definition of $f(h)$, since there are at most $(\lambda/(1-\lambda))h$ L-nodes, $f(h) \geqslant (1-\lambda/(1-\lambda))h$. This implies

$$\begin{aligned}\frac{v_C^*}{v_2^*} &\leqslant \frac{v_C^*}{f(h)}\frac{f(h)}{v_2^*}\\ &\leqslant \frac{\mathbb{E}[Y_{h,t}]}{h}\frac{f(h)}{f(h)-(p_H^2h)h}\\ &\leqslant \frac{\mathbb{E}[Y_{h,t}]}{h}\frac{(1-\lambda/(1-\lambda))h}{(1-\lambda/(1-\lambda)-p_H^2h)h}\\ &= \frac{\mathbb{E}[Y_{h,t}]}{h}\frac{1-2\lambda}{1-2\lambda-(1-\lambda)p_H^2h},\end{aligned}$$

where the second inequality uses Theorem 4. □

Using the bounds from Theorem 3 for $\mathbb{E}[Y_{h,t}]$ allows us to derive insights from Theorem 5 into the value of chains in reducing the number of unmatched H-nodes compared with using cycles only. The ratio v_C^*/v_2^* measures performance using chains compared with using two-way cycles from the outset. The smaller this ratio, the greater the marginal value of using chains. The bound (14) gives a guarantee on the expected marginal value of chains. Theorem 3(c) implies that in the case of $r=0$, even with one NDD, $\mathbb{E}[Y_{h,t}]$ can be upper bounded by $(1/p_H)\log(c/p_H)$ no matter how large h is (see the discussion after Theorem 3). Therefore, the ratio v_C^*/v_2^* is small as long as $1-2\lambda-(1-\lambda)p_H^2h$ is not close to 0. Thus when the proportion of L-nodes λ is small and $h<1/p_H^2$, the benefits of using NDDs (chains) is most substantial.

4.1. Connections to Integer Programming Formulations

In this subsection we further underscore the possibility that our bounds in Theorem 4 can be used to bound the performance of more sophisticated implementations of the Phase 2 cycle-packing algorithm. One example is the following ILP method for the clearing problem, formulated as

$$\begin{aligned}\mu_{ILP}^* := \max\ & \sum_{c\in\mathscr{C}(M)} w_cx_c\\ \text{s.t.}\ & \sum_{\{c\in\mathscr{C}(M),\,v\in c\}} x_c \leqslant 1, \quad \text{for all nodes } v, \quad (15)\\ & x_c\in\{0,1\},\end{aligned}$$

where $w(c)$ denotes the weight of cycle c, and $\mathscr{C}(M)$ denotes a set of cycles with sizes no more than M and chains of arbitrary sizes. The ILP (15) was first proposed by Abraham et al. (2007) and improved upon in subsequent studies (Dickerson et al. 2012b, 2013).

This ILP serves as the basis of the allocation scheme currently used by UNOS in clearing its exchange.

One challenge of using this method is that the number of chains in $\mathscr{C}(M)$ is exponentially increasing with the graph size and the cap on the length of chains, so the ILP can be solved in real time only for small and medium-sized graph (less than 200 nodes) and short chains (no longer than 20) (Dickerson et al. 2012b). For this reason, the current UNOS solver uses a column-generation method to strategically add potentially valuable chains and cycles into the collection $\mathscr{C}(M)$ (Dickerson et al. 2013). The next corollary states that as long as $\mathscr{C}(M)$ contains the chains that are generated in Phase 1 and all two-way cycles, then the ILP matches at least as many H-nodes as does our two-phase procedure. The lower bound for the two-phase procedure in Theorem 4 can therefore be used to lower bound the value of the ILP (15).

Corollary 3. *Let w_c denote the number of H-nodes contained in the cycle or chain c, and assume $\mathscr{C}(M)$ contains the t random walks that are generated during Phase* 1. *Suppose $r = 0$ and that cycles of at least length* 2 *are permitted in* (15) *(that is, $M \geqslant 2$). We have*

$$\begin{aligned}\mathbb{E}[\mu^*_{ILP}] &\geqslant H - v^*_C \\ &\geqslant \frac{1}{hp_H}\left((h-\ell)^+ + \frac{(1-p_Lp_H)^{|h-\ell|+1} - (1-p_Lp_H)^{\max\{h,\ell\}+1}}{p_Lp_H}\right) \\ &\quad \cdot \log\left(1 + \frac{1}{(T^0(h)+(1/4)t)p_H}\right).\end{aligned}$$

Proof. Since $\mathscr{C}(M)$ contains the t random walks that are generated during Phase 1, as well as all the two-way cycles that are used during Phase 2, any clearing generated by our two-phase procedure is actually a feasible solution to the ILP (15). According to the definition of w_c, the ILP maximizes the number of matched H-nodes. Therefore, the expected number of H-nodes matched by the two-phase procedure $H - v^*_C$ gives a lower bound for the expected optimal value of the ILP $\mathbb{E}[\mu^*_{ILP}]$. Using the lower bound (10) for v^*_C, and the strengthened upper bound (8) for $\mathbb{E}[Y_{h,t}]$ in the $r = 0$ case, we get the analytical lower bound in the statement of the corollary. □

Section 6 contains a careful numerical comparison of the two-phase procedure and an ILP-based formulation inspired by (15) to complement this basic analytical result.

5. Prioritizing Chains vs. Cycles

According to our two-phase procedure, H-L cycles (two-way cycles connecting an H-node and an L-node) are matched after allocating all random walks. However, in practice, transplant centers may myopically transplant such cycles without even submitting the patients involved to the nationwide exchange. The analysis in this section shows that allowing such two-way cycles to be matched in chains via an exchange leads to more transplants than myopically transplanting short cycles. In particular, we provide an analytical lower bound on the average number of extra transplants that can be created by prioritizing the H kidney in an H-L cycle for chain matching when the exchange pools is of medium size. Our analysis focuses on the H-L cycles only because of their analytical tractability, but our numerical studies show that there are benefits (albeit smaller) of allowing chains to take nodes that would otherwise appear in cycles of other forms (i.e., three-way cycles and H-H cycles).

Recall the set $\mathscr{C}_{HL}$ as defined after Lemma 2 in Section 4. Let $\mathscr{M} \subseteq \mathscr{C}_{HL}$ denote a maximum cardinality matching on this bipartite graph. Clearly, the set $\mathscr{M}$ depends solely on the edge set $\mathscr{C}_{HL}$, not on the edges wholly inside the H- or L-subgraphs of the random graph D. We consider two algorithms. The first is called the "Prioritize Chains," or *PriCh algorithm*, which is precisely the two-phase procedure studied up until now. This algorithm name emphasizes the fact that it prioritizes allocating *chains* in the first phase, in contrast to the second algorithm we consider. The "Prioritize Cycles," or *PriCy algorithm*, simply inverts the two phases of the PriCh algorithm. Phase 1 of the PriCy algorithm removes all two-way H-L cycles corresponding to matchings in $\mathscr{M}$ and then in Phase 2 identifies chains via random walk on the residual graph. To avoid confusion, we write PriCh.1 and PriCh.2 for Phases 1 and 2 of the PriCh algorithm and PriCy.1 and PriCy.2 for Phases 1 and 2 of the PriCy algorithm.

We still use the number of unmatched H-nodes as a performance criterion to evaluate the above two algorithms. To this end, ζ_{PriCh} and ζ_{PriCy} denote the number of unmatched H-nodes by applying the PriCh and PriCy algorithms, respectively. The next theorem shows that the PriCh algorithm always outperforms the latter. We set $m := |\mathscr{M}|$ and use the notation set in Section 3 for $Y_{h,t}$ to represent the random number of unmatched H-nodes after applying the two-phase procedure on an H-subgraph.

Theorem 6. *Consider a random kidney exchange graph $D(h,\ell,t)$ with $r = 0$. Then, the expected difference $\mathbb{E}[\zeta_{\text{PriCy}} - \zeta_{\text{PriCh}} \mid m]$ in the number of unmatched H nodes under the PriCy algorithm compared with the PriCh algorithm, conditional on there being m nodes in a maximal matching, is always positive. Moreover, we can derive lower and upper bounds on $\mathbb{E}[\zeta_{\text{PriCy}} - \zeta_{\text{PriCh}} \mid m]$ as follows*:

$$\begin{aligned}0 &< (h-m)\left(\frac{\mathbb{E}[Y_{h-m,t}]}{h-m} - \frac{\mathbb{E}[Y_{h,t}]}{h}\right) \\ &\leqslant \mathbb{E}[\zeta_{\text{PriCy}} - \zeta_{\text{PriCh}} \mid m] \leqslant \min\{m, \mathbb{E}[Y_{h-m,t}]\}. \qquad (16)\end{aligned}$$

Proof. We first prove the upper bound on $\mathbb{E}[\zeta_{\text{PriCy}} - \zeta_{\text{PriCh}} \mid m]$. By matching all two-way cycles in $\mathcal{M}$ in Phase PriCy.1, there will be $h - m$ H-nodes left for Phase PriCy.2. Because the selection of $\mathcal{M}$ is independent of edges inside the H-subgraph, on the residual H-subgraph $\mathcal{R}(m)$, each edge appears according to an independent Bernoulli distribution with mean p_H. Thus, the number of unmatched H-nodes after the two-phase procedure has the same distribution as the random variable $Y_{h-m,t}$. As a result,

$$\mathbb{E}[\zeta_{\text{PriCy}} \mid m] = \mathbb{E}[Y_{h-m,t}], \tag{17}$$

which implies

$$\mathbb{E}[\zeta_{\text{PriCy}} \mid m] - \mathbb{E}[\zeta_{\text{PriCh}} \mid m] \leqslant \mathbb{E}[\zeta_{\text{PriCy}} \mid m] = \mathbb{E}[Y_{h-m,t}]. \tag{18}$$

Alternatively, if we execute chains first, then there will be $Y_{h,t}$ H-nodes left after the two-phase procedure. Because m is the size of the maximum bipartite matching on the original graph, it must be larger than the size of the maximum bipartite matching on the graph where H-nodes have been removed by the random walk. Thus, in Phase PriCh.2, at most m H-nodes can be matched. Consequently,

$$\mathbb{E}[\zeta_{\text{PriCh}} \mid m] \geqslant \mathbb{E}[Y_{h,t}] - m. \tag{19}$$

Because $Y_{n,t}$ is increasing in n, inequality (17) and (19) imply that

$$\mathbb{E}[\zeta_{\text{PriCy}} \mid m] - \mathbb{E}[\zeta_{\text{PriCh}} \mid m] \leqslant \mathbb{E}[Y_{h-m,t}] - (\mathbb{E}[Y_{h,t}] - m) \leqslant m. \tag{20}$$

Inequalities (18) and (20) lead to the upper bound for $\mathbb{E}[\zeta_{\text{PriCy}} - \zeta_{\text{PriCh}} \mid m]$ in (16).

To derive the lower bound for $\mathbb{E}[\zeta_{\text{PriCy}} - \zeta_{\text{PriCh}} \mid m]$, we need to derive an upper bound for $\mathbb{E}[\zeta_{\text{PriCh}} \mid m]$, or equivalently, $\mathbb{E}[Y_{h,t}] - m$ because of (17). According to the PriCh algorithm, whether an H-node remains unmatched after Phase PriCh.1 is independent from whether it is covered by cycles in $\mathcal{M}$ or not. Thus, any remaining H-node has an independent probability of m/h to be covered by $\mathcal{M}$. Therefore, if we use the two-way H-L cycles in $\mathcal{M}$ to match the remaining H-nodes in Phase PriCh.2, there will be an expected number of $(1 - m/h)\mathbb{E}[Y_{h,t}]$ nodes left after Phase PriCh.2. Nevertheless, the PriCh algorithm would perform even better, because in Phase PriCh.2, the algorithm would use the maximum matching on the remaining bipartite H-L graph rather than $\mathcal{M}$ to cover the remaining H-nodes. Therefore, the expected number of unmatched H-nodes after the PriCh algorithm, $\mathbb{E}[\zeta_{\text{PriCh}} \mid m]$, is no more than $(1 - m/h)\mathbb{E}[Y_{h,t}]$. With this upper bound for $\mathbb{E}[\zeta_{\text{PriCh}} \mid m]$, together with the expression for $\mathbb{E}[\zeta_{\text{PriCy}} \mid m]$, (17), we have

$$\begin{aligned}\mathbb{E}[\zeta_{\text{PriCy}} - \zeta_{\text{PriCh}} \mid m] &\geqslant \mathbb{E}[Y_{h-m,t}] - \left(1 - \frac{m}{h}\right)\mathbb{E}[Y_{h,t}] \\ &= (h-m)\left(\frac{\mathbb{E}[Y_{h-m,t}]}{h-m} - \frac{\mathbb{E}[Y_{h,t}]}{h}\right),\end{aligned}$$

which gives the lower bound for $\mathbb{E}[\zeta_{\text{PriCy}} - \zeta_{\text{PriCh}} \mid m]$ in (16).

At last, we show that $\mathbb{E}[Y_{h-m,t}]/(h-m) - \mathbb{E}[Y_{h,t}]/h > 0$ for any $m \geqslant 1$. For all integers $n \geqslant 1$ and $t \geqslant 0$ (note that, by definition, $Y_{n,0} = n$), define

$$\Delta_n^t := \mathbb{E}Y_{n,t} - \mathbb{E}Y_{n-1,t}.$$

Then $\mathbb{E}[Y_{h-m,t}]/(h-m)$ and $\mathbb{E}[Y_{h,t}]/h$ is exactly the average of the first $h - m$ entries and the first h entries of the sequence $\{\Delta_n^t \mid n = 1, 2, \ldots\}$. By (A1) in Lemma EC.2 in Online Appendix EC.4, Δ_n^t decreases in n; therefore $\mathbb{E}[Y_{h-m,t}]/(h-m) - \mathbb{E}[Y_{h,t}]/h > 0$ for any $m \geqslant 1$. □

The lower bound in Theorem 6 shows that the expected number of transplants always increases when the nodes in H-L cycles are prioritized for use in chains. To get a more precise sense of these benefits, we propose to measure the benefit by the number of extra H-nodes that could be cleared by prioritizing each H-L cycle for use in chains. The upper bound given in Theorem 6 then shows that the average benefit for prioritizing each H-L cycle is at most one H-node (coming from the m in the upper bound term), and we provide a strengthened lower bound for the average benefit for prioritizing chains in the next theorem, for a particular class of parameter settings.

Theorem 7. *Suppose* $\lambda \leqslant 0.28$, $t \leqslant 3$, $h \geqslant (3.3/p_H) \cdot \ln(1/p_H)$, *and* $r = 0$; *then*

$$\frac{h-m}{m}\left(\frac{\mathbb{E}[Y_{h-m,t}]}{h-m} - \frac{\mathbb{E}[Y_{h,t}]}{h}\right) \geqslant \frac{0.089}{c_h}, \tag{21}$$

where $c_h = h/((1/p_H)\ln(1/p_H)) > 3.3$. *As a result, we derive upper and lower bounds on the average benefit* $(1/m)\mathbb{E}[\zeta_{\text{PriCy}} - \zeta_{\text{PriCh}} \mid m]$ *of prioritizing chains as follows*:

$$\frac{0.089}{c_h} \leqslant \frac{1}{m}\mathbb{E}[\zeta_{\text{PriCy}} - \zeta_{\text{PriCh}} \mid m] \leqslant \min\left\{1, \frac{\mathbb{E}[Y_{h-m,t}]}{m}\right\}. \tag{22}$$

The proof of Theorem 7 is involved and found in Online Appendix EC.4. The main idea is to prove that $\mathbb{E}[Y_{n,t}]$, as a function of n, is sufficiently concave (Lemmas EC.2 and EC.3), so the difference between the slopes $\mathbb{E}[Y_{h-m,t}]/(h-m)$ and $\mathbb{E}[Y_{h,t}]/h$ can be lower bounded for a given m. This will prove inequality (21). Inequality (22) then follows directly from Theorem 6.

The parameter settings we considered here ($\lambda \leqslant 0.28$, $t \leqslant 3$, $p_H \leqslant 0.1$, and $h \geqslant (3.3/p_H)\ln(1/p_H)$) cover a broad class of kidney exchange graphs. The qualitative insight implied by Theorem 7 is that the extra transplants created by using an H-L cycle in a chain rather than myopically transplanting it have a uniform lower bound that only depends on the size of the H-subgraph. Note that the machinery used in the proof of Theorem 7 (in Online Appendix EC.4) can be

adapted to derive lower bounds under other parameter settings (the choices $\lambda \leqslant 0.28$, $t \leqslant 3$, $p_H \leqslant 0.1$, and $h \geqslant (3.3/p_H)\ln(1/p_H)$ are somewhat arbitrary), but there is no closed-form expression for the value corresponding to 0.0089 in (21). If precise values for λ, t, and h are known (or that they lie in a small specified range), the bounds can be sharpened by adapting the reasoning of the proof.

One may notice that the lower bound $0.089/c_h$ provided in Theorem 6 diminishes when the graph size (and thus c_h) gets larger. This is not because our bound is less tight for large-sized graphs, but because the average number of transplants per H-L cycle prioritized for use in chains diminishes when the graph grows in size. To see this, note that $\mathbb{E}[Y_{h-m,t}]$ will remain in the order of $(1/p_H)\ln(1/p_H)$ regardless of the size of h, according to our discussions after Theorem 3. Since the total benefit is nearly a constant, the average benefit created by using an H-L cycle in a chain tends to diminish when the size of matching (m) grows with the graph size. One may also draw this conclusion by observing that the upper bound $\mathbb{E}[Y_{h-m,t}]/m$ tends to diminish for large-sized graphs.

We have assumed that m is the size of a maximum H-L bipartite matching for ease of interpretation. In fact, $\mathcal{M}$ can be any subset of $\mathcal{C}_{HL}$, as long as $\mathcal{M}$ depends on the edge set $\mathcal{C}_{HL}$ only and not on the edge distribution inside the H- nor L-subgraphs.

6. Numerical Experiments

Our two-phase procedure presented in Section 2.2 is simple by design. It handles the issue of chains in as simple a way as possible (via random assignment) so that the probabilistic implications of allocating chains is minimal. Our initial purpose for defining this procedure was to aid in analysis. This section serves as a reality check. Numerical tests give us some confidence in the strength of our analysis and how our procedure compares to other procedures implemented in the literature and in practice. Surprisingly, the performance loss from randomly assigning chains is not as great as one might expect.

This section consists of three subsections that numerically test different aspects of the two-phase procedure. The purpose of the first subsection is to examine the tightness of the bounds derived in Section 4. The second subsection provides a sense of the overall effectiveness of the two-phase procedure. We examine the performance of our two-phase procedure compared with a heuristic (due to Ashlagi et al. 2013) and an ILP implementation (based on an algorithm proposed by Dickerson et al. 2014). The third subsection examines the *order* of assigning chains and cycles, to numerically verify and extend the insights derived in Section 5.

Because of the specification of the two-phase procedure, and to fit our purposes, we conduct our numerical tests on simulated random graphs based on the model in Section 2.1 instead of the more complicated setting of a fully simulated exchange or using data based on fielded exchanges. A thorough numerical investigation undertaken with more realistic data is left for future work.

We test random graphs with 40, 60, 80, 100, 150, 200, and 300 nodes. Unless otherwise stated, we set parameter values of $p_H = 0.03$ or 0.05, $p_L = 0.45$, and $\lambda = 0.27$. We also test a variety of values for the number of NDD donors, including $t = 0$, 1, and 5, depending on the context of the comparison. These parameter choices follow those made by Ashlagi et al. (2012) to allow for straightforward comparison with their results. We repeated the experiment for different parameter values and reached similar qualitative conclusions. For each random graph specification, we randomly generate 100 graph instances, apply the algorithm to be studied, and record the number of unmatched H-nodes at termination. Values are averaged across these observations to yield average performance measures of our algorithm and various comparisons. The performance metric we favor is the number of unmatched H-nodes in the graph (unless otherwise stated). The smaller this number, the better the performance.

6.1. Tightness of Theoretical Bounds

In our first numerical experiment, we compare the actual performance of the two-phase procedure as defined in Section 2.2 with bipartite matching in Phase 2 on simulated random graphs (in the third column of Table 1, with the theoretical upper bounds for v_C^* derived in Theorem 4). This will give a sense of the tightness of our theoretical analysis in comparison to simulated instances. We calculate upper bounds using (10) and record these values in the fourth column of Table 1. The simulated values are within 15% of the upper bounds for most instances, although the upper bound weakens as the number of nodes and NDDs increase. If we compare across the different values for t, we find a significant reduction in the number of unmatched H-nodes when chains are allowed ($t > 0$). The fifth and sixth columns of Table 1 provide a comparison of simulated values for the ratio v_C^*/v_2^* and our bound in Theorem 5. We used an optimal blossom algorithm to calculate the optimal matching with two-way cycles only, which gives v_2^*. We do not conduct the tests when there were no NDDs, in which case comparing v_2^* and v_C^* makes no sense. We conclude that our analytical bounds are sufficiently tight to have confidence in the strength of our analysis.

6.2. Overall Performance

In a second set of experiments, we compare the performance of our two-phase procedure (defined in Section 2.2) with more sophisticated clearing algorithms,

Table 1. Tightness of Bounds Derived in Section 4

No. nodes	No. NDDs	v_C^* via simulation	Bound for v_C^*	v_C^*/v_2^* via simulation	Bound for v_C^*/v_2^*
40	0	26.67	27.08		
40	1	25.53	25.79	0.98	1.00
40	5	20.88	21.94	0.80	0.85
60	0	37.09	37.76		
60	1	34.82	35.23	0.98	0.99
60	5	27.50	29.19	0.78	0.82
80	0	47.91	48.83		
80	1	43.20	45.68	0.95	1.00
80	5	33.41	36.66	0.74	0.82
100	0	56.66	58.01		
100	1	50.82	54.10	0.97	1.00
100	5	36.75	42.08	0.70	0.80
150	0	78.36	80.85		
150	1	63.91	68.44	0.91	0.99
150	5	38.81	44.25	0.55	0.65
200	0	100.22	102.88		
200	1	71.65	78.47	0.84	0.96
200	5	37.22	45.55	0.43	0.56
300	0	142.67	145.47		
300	1	66.85	85.59	0.61	0.85
300	5	27.72	44.56	0.25	0.44

Notes. Table entries are the number of unmatched H-nodes remaining. A lower number indicates better performance. In the column labels, v_C^* and v_2^* denote the expected number of unmatched H-nodes after applying the two-phase algorithm and the optimal two-way cycle-packing algorithm to the random graph, respectively. Accordingly, v_C^*/v_2^* thus gives the performance ratio between algorithms that use both chains and two-way cycles or use two-way cycles only. The theoretical upper bounds for v_C^* and v_C^*/v_2^* are provided in (10) and (14), respectively.

Table 2. Performance of Two-Phase Procedure Compared with Heuristic in Ashlagi et al. (2012) and the ILP with Chains Generated as Random Walks

No. of nodes	No. of NDDs	Two-phase procedure	Ashlagi et al. (2012)	ILP
40	0	23.46	23.81	20.22
40	1	22.20	22.80	15.23
40	5	18.52	19.97	13.18
60	0	28.68	30.79	25.42
60	1	26.47	27.21	16.16
60	5	22.91	21.98	13.41
80	0	33.54	36.59	26.93
80	1	31.17	29.68	15.95
80	5	24.17	21.87	12.72
100	0	36.11	40.67	27.00
100	1	32.75	29.91	14.72
100	5	24.46	19.13	11.57
150	0	39.66	Unavailable	19.60
150	1	34.41	Unavailable	10.29
150	5	22.67	Unavailable	7.57
200	0	40.37	Unavailable	8.40
200	1	32.19	Unavailable	5.26
200	5	19.37	Unavailable	3.24
300	0	35.68	Unavailable	0.18
300	1	24.37	Unavailable	0.13
300	5	12.92	Unavailable	0.12

Notes. Table entries are the number of unmatched H-nodes remaining. A lower number indicates better performance.

such as the heuristic algorithm by Ashlagi et al. (2012) and the ILP with chains generated as random walks. We find that our two-phase algorithm, which uses a heuristic algorithm to match the three-way and two-way cycles in the residual graph in Phase 2, achieves comparable performance as the heuristic algorithm of Ashlagi et al. (2012). We also identify parametric settings in which the two-phase algorithm is competitive with ILP, which can be regarded as a proxy for the optimal algorithm. The gap between the two-phase algorithm and the ILP is narrowed for larger graph sizes, greater numbers of NDDs, denser exchange graphs, and larger proportions of L-nodes (relative to H-nodes). We also find that the above gap widened when cycles are not permitted at all, suggesting that the random walk is actually not good at picking the longest chains. This, however, does not result in a large gap when cycles are used. This can be explained by the fact that the random walk of Phase 1 preserves the density of the edge distribution in the residual graph, an advantage for cycle matching in Phase 2.

Our first comparison is with results given in table 5 of Ashlagi et al. (2012), which records the number of matched H-nodes after running their heuristic algorithm. Their matching algorithm uses three-way cycles, and so to make the comparison fair, we also allow three-way cycle matching in the Phase 2 of the algorithm. The comparison is displayed in Table 2, where the third column records the average number of unmatched H-nodes using Phase 1 and Phase 2 of our procedure across 100 randomly generated graphs in each setting, and the fourth column records the number of unmatched H-nodes inferred by table 5 of Ashlagi et al. (2012). Note that Ashlagi et al. (2012) only report values for exchanges up to 100 nodes in size. These results are encouraging. Although the number of unmatched H-nodes from our procedure increases relative to the Ashlagi et al. (2012) heuristic, the performance is nonetheless comparable.

Next, we attempted to compare our algorithm to an optimal clearing algorithm. However, as the problem is NP-hard (Abraham et al. 2007), we cannot expect that the optimal solution is readily computed, particularly as the number of nodes grows larger. Existing heuristics typically tightly cap the length of chains. If chains are long (which does happen in practice; see Sack 2012) the generation of *many* variables in an ILP formulation such as (15) is required. We worked with the optimization code provided by Dickerson et al. in May 2016 (https://github.com/johndickerson/kidneyexchange) that accompanied the publication of Dickerson et al. (2014). When we removed the chain

cap, the code was unable to find optimal solutions on larger instances in a reasonable run time. In response, we developed a modified version of the Dickerson et al. code that uses random walk to sample a (reasonably sized) set of chains that enter an ILP formulation as variables (inspired by personal communication with John P. Dickerson et al. (2014)). The generated random walks prioritize H-nodes as in our two-phase procedure. Following standard practice, we capped the length of cycles at 3 because of the limitation that cycles must be transplanted simultaneously and hence logistically infeasible at lengths greater than 4. This makes the number of variables associated with cycles also manageable. In our simulations, we capped the number of chain variables at 100,000, meaning that the ILP formulation worked with at most 100,000 chains generated as random walks and all cycles up to length 3.

The performance of this modified ILP algorithm, which uses elements of integer programming and random walks, is reported in the final column of Table 2. We believe this hybrid algorithm gives a good approximation to an optimal clearing algorithm. Indeed, for graphs of 300 nodes, it leaves almost no unmatched H-nodes even without using any chains. The gap between the two-phase algorithm and the ILP stays below 30 H-nodes across all scenarios.

Our goal is not to advocate that the two-phase procedure as stated in Section 2.2 be used in practice in its simple form; instead, our motivation is to show that it may be useful to incorporate a very simple allocation of chains via random walk in the first stage into more elaborate methods, such as the ILP. Using random walks can greatly speed the computational burden of finding optimal clearings. Our simple two-phase procedure runs very quickly with less than 0.1 seconds for all instances, while the more elaborate ILP formulation can take up to 1,000 seconds for instances with larger graph sizes.

An interesting question is under what parameter setting does our two-phase procedure come close to being optimal. We use the proportion of unmatched H-nodes instead of the number of H-nodes as a performance measure, as it provides a fair comparison across instances with different graph sizes. Figure 6 illustrates that the performance of the two-phase procedure improves as (a) the total number of nodes get larger, (b) the number of NDD nodes t get larger, (c) p_H gets larger, and (d) the proportion of L-nodes λ get larger. In (a), we find that when the graph size expands, the gap between the two-phase procedure and the ILP reduces quickly. For graphs with more than 200 nodes, our algorithm matches more than 80% of the H-nodes, which shows that the two-phase algorithm is most competitive when the graph size is sufficiently large. In (b), we notice that increasing the number of NDDs steadily improves the performance of the two-phase algorithm, but not the ILP. The intuition is that the ILP is always able to identify very long chains, so one NDD often suffices; in our algorithm, however, we cannot expect a single random walk to match most H-nodes, so adding extra NDDs (more random walks) helps. In (c) and (d), we find that the performance of both our algorithm and the ILP improve, while their gap shrinks, when the graph contains more edges in the H-subgraph or contains a larger proportion of L-nodes. In these cases, the cycle-matching phase becomes relatively more important, which diminishes the suboptimality introduced in Phase 1 of the two-phase procedure.

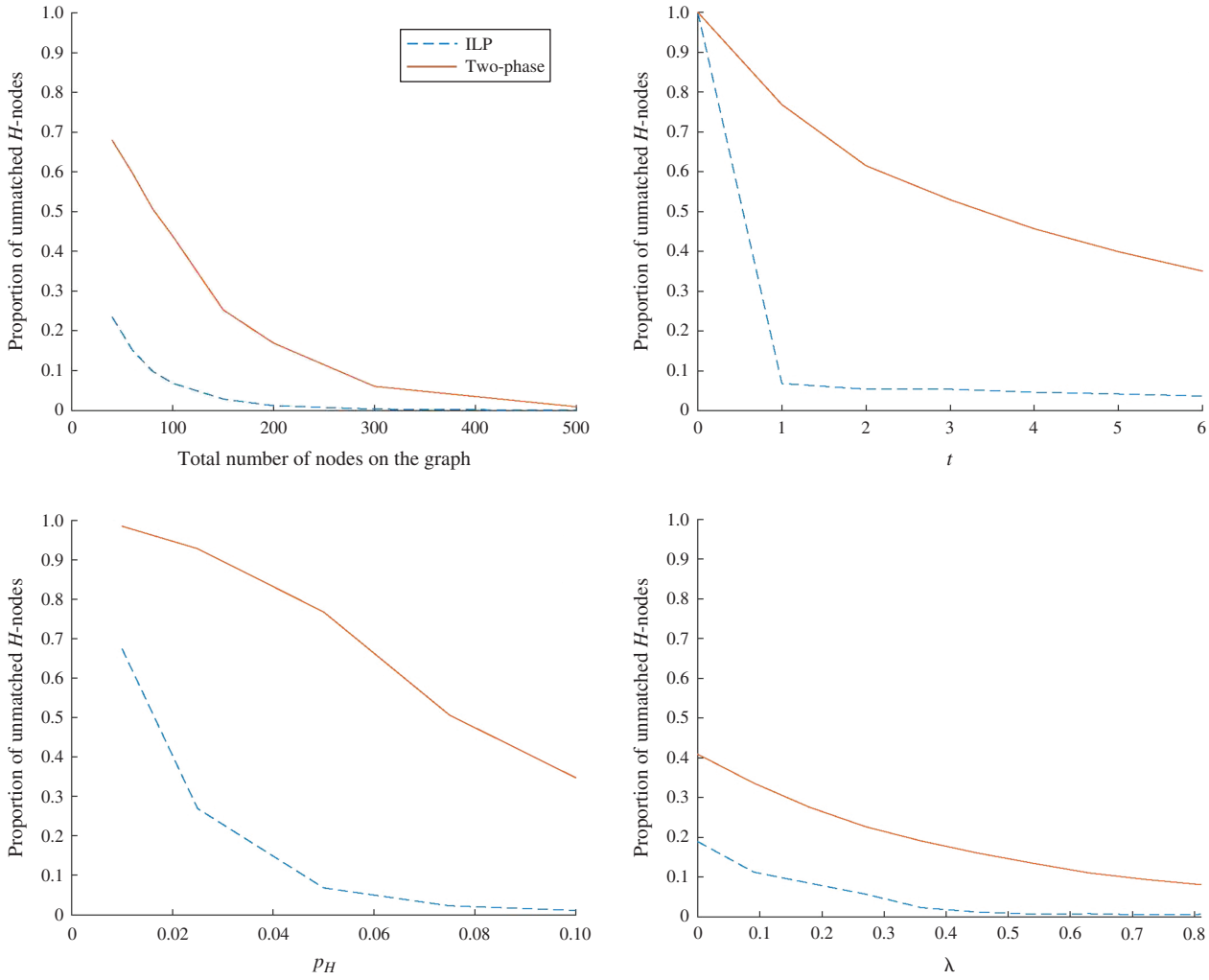
Thinking more systematically, observe that our two-phase procedure cannot perform as well as the ILP for two separate reasons. First, the ILP formulation can choose the best combination of cycles and chains out of a large set of generated random walks, whereas our procedure can only optimize cycle matching *after* chains have been removed. Second, the chains generated by the random walks cannot, in general, match as many nodes as those generated by the ILP. To further understand the main reason that leads to the performance gap, we consider a special case when the graph contains only H-nodes and no cycles are allowed. We then compare the number of nodes that are covered by chains generated by random walk versus our ILP formulation, and we summarize the results in Table 3. We set $p_H = 0.05$ and $\lambda = 0$, as there is no need to consider L-nodes for this experiment.

Table 3 shows that the gap could be as large as 50 H-nodes between the random walk and the ILP. However, in Table 2 we observe that the gap is less than 30 in the presence of cycles. One explanation is that although random walks are a suboptimal way to allocate chains, in the presence of cycles, this suboptimality is lessened. Optimally allocating chains may preclude an efficient allocation of cycles in the second stage, but this effect is weakened under random walk allocation. As discussed above, random walk allocation in the first phase maintains the probabilistic density of the residual graph, enhancing cycle matching. Hence, we believe the major loss of optimality from our procedure is not that it may choose a chain that is too short but that it is unable to look forward to assess the impact of choosing a chain on the cycle formation phase. This returns us to the discussion of the impact of prioritizing chains versus cycles that we initiated in Section 5.

6.3. Prioritizing Cycles Before Chains

In Section 5 we showed that if we must prioritize either chains or cycles in our two-phase procedure, it is preferable to prioritize chains. These results were analytical and focused attention on H-L cycles for purposes of tractability. In this section we reinforce numerically, and in greater generality, the same general conclusion by allowing a wider variety of cycles.

Figure 6. (Color online) An Illustration of How the Performance of the Two-Phase Procedure Compared with an ILP Formulation Under Various Parameter Changes



Notes. In each panel, we compare the proportion of unmatched H-nodes by varying the total number of nodes ($h + l$), t, p_H, and λ. When one parameter is changing, the other parameters are set to their default values: $h + l = 100$, $t = 1$, $p_H = 0.05$, and $\lambda = 0.27$.

Figure 7 illustrates the benefit of prioritizing chains over prioritizing cycles. The vertical axis captures the difference in the number of nodes, and the horizontal axis is the net increase in the number of H nodes matched when prioritizing chains compared with prioritizing cycles. That is, the vertical axis captures simulated values of $\mathbb{E}[\zeta_{\text{PriCy}} - \zeta_{\text{PriCh}}]$. Larger values on the vertical axes correspond to increased benefits of prioritizing chains. The figure illustrates three curves. The first is when bipartite matching between H-L-nodes is used in the PriCy and PriCh algorithms—the setting discussed in Section 5. This case is denoted as "H-L" in the figure, meaning the cycle formation phase employs H-L cycles only. The second curve still prioritizes H-L cycles but in the PriCy algorithm adds an additional step after chains are formed to use three-way matching in the residual graph that remains. This case is denoted as "H-L then 3-way" in the figure. This is a first attempt to "improve" the PriCy algorithm beyond what was studied in Section 5. Observe that the improvement is significant but still performs worse than the PriCh algorithm. Finally, the third curve is a second attempt to improve the PriCy algorithm, where instead of only prioritizing H-L cycles, all cycles involving up to three patients are prioritized. This case is denoted as "3-way" in the figure. The improvement is dramatic over the previous two curves but, somewhat surprisingly, still reveals a small benefit for prioritizing chains over cycles. Figure 7 is drawn for the case of a single NDD; we did similar simulations for three and five NDDs, and the basic pattern was similar.

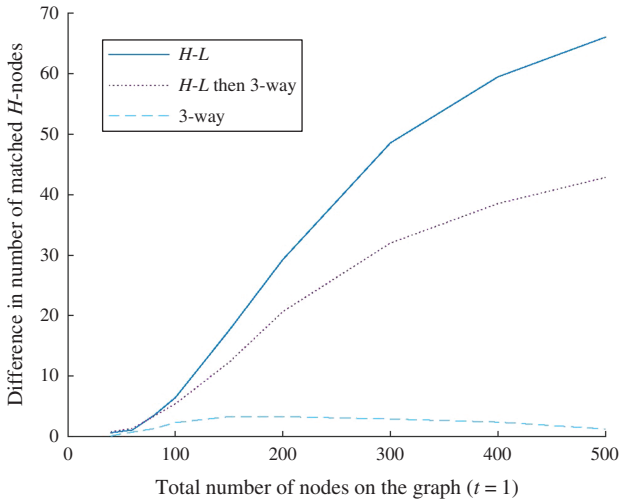
One implication of these results is that initially prioritizing H-L cycles can undermine subsequent matching in chains and additional cycles, and this impact is lessened when a wider variety of cycles are prioritized. This has practical relevance since individual hospitals

Table 3. Performance of Two-Phase Compared with an ILP Formulation When There Are No Cycles

No. of nodes	No. of NDDs	Two-phase procedure	ILP
40	1	33.84	20.40
40	5	22.87	11.52
60	1	48.15	16.90
60	5	27.77	10.70
80	1	56.23	16.95
80	5	28.88	10.67
100	1	61.72	16.51
100	5	11.15	30.61
200	1	67.34	16.38
200	5	31.05	7.58
300	1	68.95	12.39
300	5	31.12	6.45

Notes. Table entries are the number of unmatched H-nodes remaining. A lower number indicates better performance.

Figure 7. (Color online) Comparing the Benefits of Prioritizing Chains vs. Prioritizing Cycles, with $\lambda = 0.27$, $p_H = 0.05$, and $t = 1$



are more likely to transplant H-L cycles before declaring their patients in an exchange than they are any other transplant combination involving an H-node. A more general implication one can infer from this is that myopically transplanting the easiest cases has the greatest negative impact on the quality of clearing in an exchange. Indeed, further numerical investigation (not reported here) suggests that the more L-nodes in the chains that are prioritized (for instance, H-L or H-L-L compared with H-H-L), the worse is the overall performance.

Another observation is that the loss of transplants created by matching H-L cycles increases significantly as the graph size expands. This is in line with the lower bound in Theorem 7 that increases with the size of the graph. However, since we have used a sophisticated heuristic algorithm to match the three-way cycles, the loss incurred by prioritizing up to three-way cycles stays low irrespective of the graph size. The reason is that a sophisticated heuristic algorithm will intelligently use H-H-H or H-H-L cycles, in which H-nodes take a larger proportion. As suggested by Figure 7, removing those cycles first do not result in a big loss. This underscores how the impact of the order of prioritizing chains versus cycles is *not* robust when attention is isolated to H-L cycles but is robust when up to three-way cycles are considered, provided that a sophisticated cycle-matching heuristic is used.

7. Directions for Future Work

In this paper we have developed a nonasymptotic approach to analyzing kidney exchange graphs that complements previous work that relies on asymptotic analysis. We demonstrate the power of this approach by providing analytical performance bounds on a two-phase procedure for matching donors and recipients, and we demonstrate how these bounds allow us to analytically show the benefit of chains in "medium-sized" (that is, nonlimit) graphs.

We developed our approach in the stylized setting introduced by Ashlagi et al. (2012) with one additional restriction in our procedure to facilitate analysis. The chains in our two-phase procedure consist entirely of H-nodes (initiated, of course, by an NDD donor). We did this to maintain the stochastic independence structure of residual graphs that was leveraged at several points in our proofs. Nonetheless, extending to "mixed chains" of both H- and L-nodes may be approachable by adjusting our methodology and is a topic of further investigation.

Of course, there are several important assumptions inherent in the random graph model of Ashlagi et al. (2012) itself. Discussion of the validity of these assumptions and the possibility of extension is well documented (see, for instance, Ashlagi et al. 2012, 2013; Dickerson et al. 2012b, 2013). A test bed for the power of our nonasymptotic approach is to systematically attempt to relax certain assumptions and see what tractability remains. Since our approach has important distinctions with the standard asymptotic methods, it is conceivable that we can achieve further generality in ways that are not amenable to other methods. Some promising avenues include embedding random walks in a dynamic setting of kidney exchange within an evolving patient and donor base. We feel that the memoryless properties of random walks should prove useful in a dynamic setting.

Finally, although we introduced our procedure primarily for analytical investigation, we feel it is useful to comment on the practicality of the approach. Allocating organs on the basis of random walks may strike practitioners and patients alike as somewhat

arbitrary and unfair. However, there is some flexibility in our procedure that could make it more palatable in practice. One can specify a *priority* by which to process the nodes. This priority could increase the likelihood that a given patient (potentially a very deserving one) could get a kidney sooner than others. One way of doing this is to evaluate the potential of each node on the kidney exchange graph, as proposed in Dickerson and Sandholm (2015). Of course, a benefit of the random walk approach is its scalability and ease of implementation, which might complement some of the more sophisticated algorithms proposed by other researchers.

Acknowledgments

The authors thank two anonymous reviewers, the associate editor, and the area editor for their valuable comments. They also thank Yutong Liu, Tian Xie, Yang Lv, Qiaowen Guo, and Shangyou Wu for their technical assistance and John Dickerson and Jiawei Zhang for enlightening conversations. Part of this research was conducted at the Center for Management Science and Information Analytics at the Shanghai University of Finance and Economics. The authors thank the Center for its generosity in facilitating their collaboration.

References

Abraham DJ, Blum A, Sandholm T (2007) Clearing algorithms for barter exchange markets: Enabling nationwide kidney exchanges. *Proc. 8th ACM Conf. Electronic Commerce* (ACM, New York), 295–304.

Ashlagi I, Jaillet P, Manshadi VH (2013) Kidney exchange in dynamic sparse heterogenous pools. *Proc. 14th ACM Conf. Electronic Commerce* (ACM, New York), 25–26..

Ashlagi I, Gamarnik D, Rees MA, Roth AE (2012) The need for (long) chains in kidney exchange. Technical report, National Bureau of Economic Research, Cambridge, MA.

Ashlagi I, Gilchrist DS, Roth AE, Rees MA (2011) Nonsimultaneous chains and dominos in kidney-paired donation—Revisited. *Amer. J. Transplant.* 11(5):984–994.

Dickerson JP (2014) Personal communication with Yichuan Ding by email.

Dickerson JP, Sandholm T (2015) FutureMatch: Combining human value judgments and machine learning to match in dynamic environments. *AAAI Conf. Artificial Intelligence* (*AAAI*) (Association for the Advancement of Artificial Intelligence, Menlo Park, CA), 622–628.

Dickerson JP, Procaccia AD, Sandholm T (2012a) Dynamic matching via weighted myopia with application to kidney exchange. *Proc. 2012 Internat. Joint Conf. Artificial Intelligence* (Association for the Advancement of Artificial Intelligence, Menlo Park, CA), 1340–1346.

Dickerson JP, Procaccia AD, Sandholm T (2012b) Optimizing kidney exchange with transplant chains: Theory and reality. *Proc. 11th Internat. Conf. Autonomous Agents Multiagent Systems*, Vol. 2 (International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC), 711–718.

Dickerson JP, Procaccia AD, Sandholm T (2013) Failure-aware kidney exchange. *Proc. 14th ACM Conf. Electronic Commerce* (ACM, New York), 323–340.

Dickerson JP, Procaccia AD, Sandholm T (2014) Price of fairness in kidney exchange. *Proc. 2014 Internat. Conf. Autonomous Agents Multi-Agent Systems* (International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC), 1013–1020.

Erdős P, Rényi A (1959) On random graphs I. *Publicationes Math. Debrecen* 6:290–297.

Gentry SE, Montgomery RA, Swihart BJ, Segev DL (2009) The roles of dominos and nonsimultaneous chains in kidney paired donation. *Amer. J. Transplant.* 9(6):1330–1336.

Hall P (1935) On representatives of subsets. *J. London Math. Soc.* 10(1):26–30.

Melcher ML, Leeser DB, Gritsch HA, Milner J, Kapur S, Busque S, Roberts JP, et al. (2012) Chain transplantation: Initial experience of a large multicenter program. *Amer. J. Transplant.* 12(9): 2429–2436.

Roth AE, Sönmez T, Ünver MU (2007) Efficient kidney exchange: Coincidence of wants in markets with compatibility-based preferences. *Amer. Econom. Rev.* 97(3)828–851.

Sack K (2012) 60 lives, 30 kidneys, all linked. *New York Times* (February 18), http://www.nytimes.com/2012/02/19/health/lives-forever-linked-through-kidney-transplant-chain-124.html.

Toulis P, Parkes DC (2011) A random graph model of kidney exchanges: Efficiency, individual-rationality and incentives. *Proc. 12th ACM Conf. Electronic Commerce* (ACM, New York), 323–332.

Ünver MU (2010) Dynamic kidney exchange. *Rev. Econom. Stud.* 77(1):372–414.

Woodle ES, Daller JA, Aeder M, Shapiro R, Sandholm T, Casingal V, Goldfarb D, Lewis RM, Goebel J, Siegler M (2010) Ethical considerations for participation of nondirected living donors in kidney exchange programs. *Amer. J. Transplant.* 10(6): 1460–1467.

Yichuan Ding is an assistant professor in the Operations and Logistics Division, Sauder School of Business, University of British Columbia. His research interests include optimization and queueing theory, as well as their applications in public sector.

Dongdong Ge is a professor at the Research Institute for Interdisciplinary Sciences at the Shanghai University of Finance and Economics. His research interests mainly lie in large-scale optimization theory and applications.

Simai He is a professor at the Research Institute for Interdisciplinary Sciences at the Shanghai University of Finance and Economics. His research interests include optimization and game theory, as well as applications in supply chain management.

Christopher Thomas Ryan is an associate professor of operations management at the Booth School of Business at the University of Chicago where he teaches service operations management.