Harvard CS 1360, Spring 2026
Instructor: Ariel Procaccia
Scribe: Jessica Chen, Jakob de Raaij

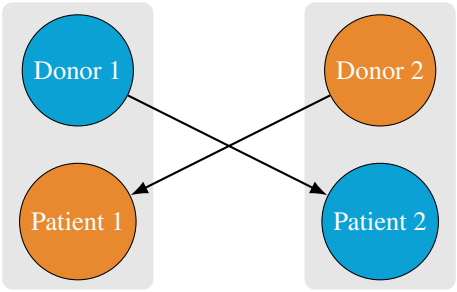
Kidney Exchange

—

Lecture 14

Kidney failure is a serious medical issue. The preferred treatment is a kidney transplant from either a deceased or live donor (as most humans are born with two functioning kidneys, many times one can be donated). However, the donor has to be blood-type and tissue-type compatible, which may prevent willing donors from being able to donate their kidney to a designated patient.

In some cases, a patient may have a willing but incompatible donor, such as a sibling or spouse with a different blood type. However, if there are two such patient-donor pairs, in which the patients and donors are cross-compatible, then the two transplants can still take place in a pairwise exchange.



For example, assume donor 1 is willing to donate to patient 1 (say, they are a sibling), and donor 2 is willing to donate to patient 2. However, it may be the case that donor 1 is only compatible with patient 2 (blue), while donor 2 is only compatible with patient 1 (orange). If the two patient-donor pairs do a pairwise exchange, both patients can receive a kidney.

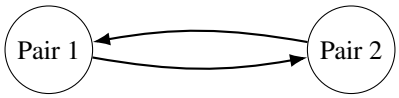
While the example above shows a pairwise exchange, we could generally have exchanges along longer cycles. However, there are practical limitations: If the donations happen one after another, a later donor may decline to donate once 'their' patient has received a kidney, before they give away their kidney. It is not legally possible to prevent this behavior, so the operations in practice must take place simultaneously. Typically, cycles of length three are the limit.

The problem of deciding how to best do these cyclic donation exchanges is of significant importance, with algorithms playing a big role in practice. As of March 25, 2025, there were 90,489 patients waiting for a kidney transplant in the United States, and the number of patients in the queue continues to grow.[1]

1 Optimal Kidney Exchange

Definition 1 (Kidney Exchange Graphs)**.** Given is a directed graph $G = (V, E)$, where V is the set of donor-patient pairs and there is an edge from u to v if the donor of u is compatible with the patient of v.

The example from above, as a kidney exchange graph, becomes



Finding the kidney exchanges that maximize the number of patients that receive a kidney, subject to donors only be willing to give their kidney if 'their' patient receives a kidney, is a version of the cycle cover problem.

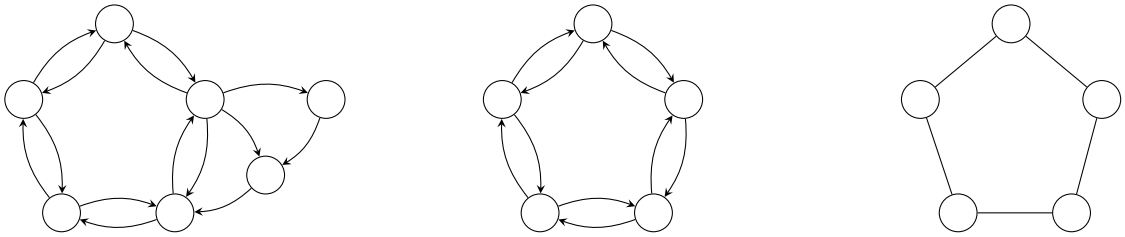
Definition 2 (Cycle Cover)**.** For any integer L, the *L-cycle cover problem*, given a directed graph G and an integer k, is to find a collection of vertex-disjoint cycles in G, all of length at most L, that cover at least k vertices.

[1] Live transplant database

Each vertex represents a patient-donor pair and each edge represents a possible donation. The goal is to find the collection of disjoint, directed cycles of some maximum length L, corresponding to (realized) donations. The cycles must be vertex-disjoint, as a donor cannot give a kidney twice.

Theorem 1. *For any constant $L \geq 3$, L-cycle cover is **NP**-complete.*

Unfortunately, this implies that unless **P** = **NP**, no polynomial-time algorithm can solve L-cycle cover for any constant $L \geq 3$. In assignment 4, we consider the case where there's no upper-bound on the cycle length, $L = \infty$, and will show that this problem is in P. Furthermore, the case $L = 2$ is also tractable: If we only focus on cycles of length 2, we only need to consider pairs of vertices that are connected by an edge in both ways (for example, instead of the graph below on the left, it suffices to consider the graph in the middle). We can replace each such 2-cycle with an undirected edge that represents a pairwise exchange (the graph on the right). Then, we can find a 2-cycle cover including the maximum number of vertices by finding a maximum cardinality matching in the new, undirected graph, which we can do in polynomial time.



In practice, many modern kidney exchanges use a cycle length upper bound $L = 3$. Even though it is theoretically intractable, we can solve the problem for small $L \geq 3$ in practice using integer linear programming.

We let C_L be the set of all cycles of length at most L. For each cycle $c \in C_L$ of length $l_c \leq L$, let the variable $x_c \in \{0, 1\}$ be the indicator variable for cycle c being included in the L-cycle cover. An L-cycle cover including a maximum number of vertices can be found with the ILP

$$\begin{aligned} \max \quad & \sum_{c \in C_L} x_c \ell_c \\ \text{s.t.} \quad & \forall v \in V : \sum_{c \in C_L:\ v \in c} x_c \leq 1, \\ & \forall c \in C_L : \ x_c \in \{0, 1\}. \end{aligned}$$

The objective corresponds to the vertices included in cycles in the L-cycle cover, namely l_c if c is part of the cover. The first set of constraints ensures that each vertex is included in at most one cycle in the cover.

The United Network of Organ Sharing (UNOS) has been using these ILP-based approaches since the inception of the exchange in 2002. The algorithms were originally developed by Tuomas Sandholm and his group at Carnegie Mellon University. Much follow-up work has further optimized and extended this approach, for example to allow for *altruistic donors* that are willing to give a kidney without being coupled with a patient looking for a kidney. This allows for donation chains (in addition to cycles), further complicating the computational problem.

2 Strategic Kidney Exchange

Initially, kidney exchanges were carried out by individual hospitals. Today, there are nationally organized exchanges, with participating hospitals having little interaction with the kidney exchange mechanism. It was observed that many hospitals do kidney exchange among easy-to-match patient-donor pairs internally and only report patients that are difficult to match to the national network. This may be strategically advisable for a hospital seeking to maximize the number of *their own* patients that get matched, but may create inefficiency in the bigger picture — if some patient-donor pairs aren't reported to the national database, we may not be able to find the optimal kidney exchange. Thus, the goal is to incentivize hospitals to enroll all their patient-donor pairs to the national network.

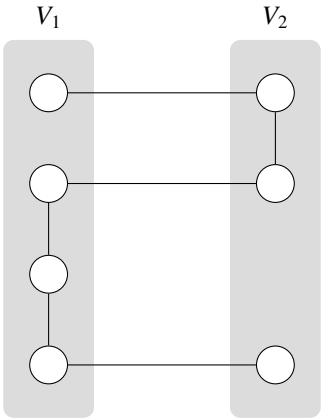
We now formalize this. For simplicity, we will consider the case $L = 2$, where only pairwise exchanges are allowed. Thus, we'll only consider undirected edges in the kidney exchange graph.

Definition 3 (Strategic Kidney Exchange)**.** An instance of *kidney exchange with multiple hospitals* consists of an undirected (kidney exchange) graph $G = (V, E)$ and a partition of the patient-donor pairs V into *hospitals* $V_1, ..., V_n$. A *kidney exchange mechanism* receives as input such an instance of kidney exchange with multiple hospitals and returns a matching in G.

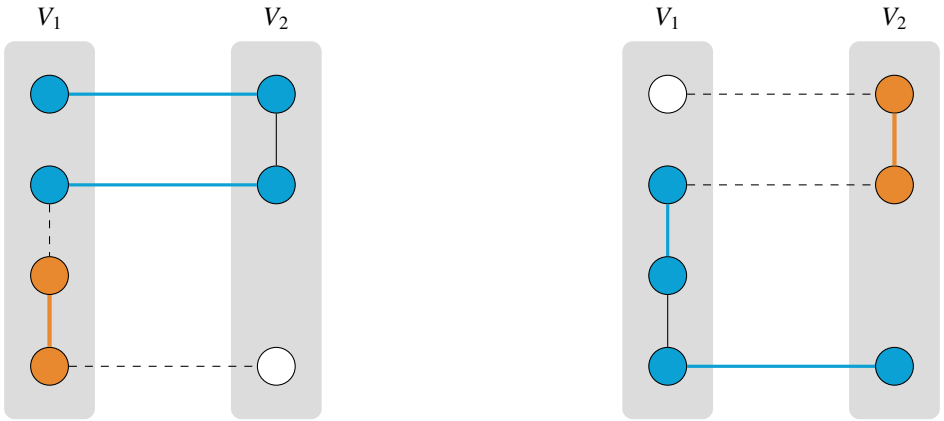
Each hospital V_i is a player i, with their utility being the number of their own vertices (V_i) that have been matched. The strategies of each player are which vertices they reveal to the mechanism and which vertices they don't (i.e., which patients they report to the national program). For any two reported vertices connected by an edge, this edge is known — the hospitals cannot strategically conceal medical compatibility.

Our objective is to maximize the utilitarian social welfare, i.e., the size of the matching (equivalently, the number of patients that receive a donation), subject to hospitals reporting patients strategically and selfishly. We say that a kidney exchange mechanism is *strategyproof* if it is a dominant strategy for every hospital to report all their vertices. That is, it is never beneficial for a hospital to conceal patients.

Unfortunately, no strategyproof mechanism can always maximize utilitarian social welfare. To see this, consider the following kidney exchange graph with two hospitals



Since there are seven vertices, either a vertex in V_1 or a vertex in V_2 must remain unmatched. However, either player can ensure that all their vertices are matched by strategically not reporting certain vertices. As shown in the figure on the left below, if player 1 conceals their bottom two vertices, any socially optimal mechanism has to match the top two pairs of vertices across hospitals. Player 1 can match the two concealed vertices internally and have all four of their vertices matched. As shown in the figure on the right below, if player 2 conceals the top two vertices, any socially optimal mechanism has to match the remaining vertices as shown in blue. Player 2 can match the two concealed vertices internally and have all three of their vertices matched. Thus, no matter what kidney exchange matching a mechanism returns on the graph with full information, one of the two players would be better off concealing some of their vertices.

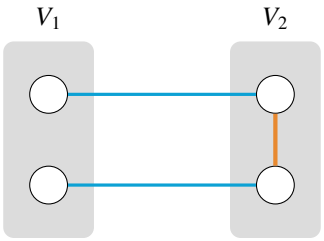


Given that we cannot achieve a socially optimal outcome while also being strategyproof, we now explore if we can at least approximate the social optimum. Consider the following algorithm:

Algorithm 1 $\textsc{Match}_{\{\{1\},\{2\}\}}$ Mechanism

1: Consider all matchings that maximize the number of internal edges (between two vertices of the same player) that are included
2: Among these, return the matching of maximum cardinality.

The $\textsc{Match}_{\{\{1\},\{2\}\}}$ mechanism is exactly a $1/2$-approximation to the social optimum. Since it always returns an inclusion maximal matching — no edge can be added without removing another edge — we know that it is a $1/2$-approximation, by the same argument as for the greedy algorithm last lecture: For any edge in the optimal matching, at least one of its two vertices must be matched in any inclusion maximal matching. To see that it is no better than a $1/2$-approximation, consider the graph below. The social optimum would be to match all vertices, shown in blue, while the $\textsc{Match}_{\{\{1\},\{2\}\}}$ mechanism only matches the two vertices in V_2, shown in orange.

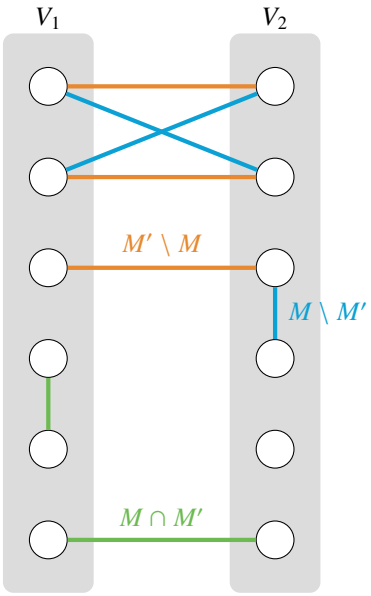


While a worst-case approximation of $1/2$ to the optimum may seem too far from optimal, it was observed that in practice, this mechanism reaches about a 90% approximation to the social optimum.

Theorem 2. *For two players, the* $\textsc{Match}_{\{\{1\},\{2\}\}}$ *mechanism is strategyproof.*

Proof. We will show that the mechanism is strategyproof for player 1, the same then follows for player 2 by symmetry. Let M be the matching output by the mechanism when player 1 reports all their vertices truthfully, and let M' be the matching when player 1 hides some vertices and matches some internally. We want to argue that the matching M is at least as good (i.e., has at least as many matched vertices) for player 1 as M', the matching player 1 can obtain from misreporting, so that player 1 does not benefit from hiding some of their vertices.

To do this, we consider the symmetric difference $M \Delta M'$, which consists of all edges in M but not in M' (blue in the example below) and all edges in M' but not in M (orange in the example below). We ignore edges that are included in both M and M' (green in the example below).



This symmetric difference $M \Delta M'$ consists of vertex-disjoint *alternating* paths and cycles. That is, in every path or cycle in $M \Delta M'$, edges from $M \setminus M'$ (blue) and $M' \setminus M$ (orange) alternate. This holds true since given two edges incident to the same vertex, at most one of the edges can be part of the same matching, since no vertex can be used twice in a matching.

We now show M is at least as good as M', i.e., $|M| \geq |M'|$, for every component (every path and cycle) of the symmetric difference separately. On cycles, M and M' match the same set of vertices, so Player 1's utility is the same; we therefore focus on paths.

Consider a single path in the symmetric difference $M \Delta M'$, and denote its edges in M (blue above) by P and its edges in M' (orange above) by P'. For $i, j \in \{1, 2\}$, we let P_{ij} (and P'_{ij}) be all edges in P (in P') from a vertex of player i to a vertex of player j,

$$P_{ij} = \{(u, v) \in P : u \in V_i, v \in V_j\}, \qquad P'_{ij} = \{(u, v) \in P' : u \in V_i, v \in V_j\}.$$

Thus, P_{11} are the edges in P *internal* to player 1, P_{22} are the edges in P internal to player 2, and P_{12} are the *external* edges in P between a vertex of player 1 and player 2; analogously for P'.

We know that given full information about the graph, the mechanism maximizes the number of internal edges for each player. This applies to every path separately: If we had some path along which M' has more edges in $P'_1 1$ than $P_1 1$, we could switch the edges from M to M' and get more internal edges in M —a contradiction to the mechanism maximizing the number of internal edges. Thus, it always holds that $|P_{11}| \geq |P'_{11}|$. Given this weak inequality, there are two cases:

Case 1: $|P_{11}| = |P'_{11}|$. Since player 2's vertices are unchanged, the mechanism has access to the same vertices and edges within V_2 for both M and M'. Hence, it holds that $|P_{22}| = |P'_{22}|$.

Recall that the mechanism selects a matching of maximum cardinality subject to maximizing the number of internal edges. M and M' both are constraint to have the same number of internal edges, but M had at least as much freedom in picking which internal edges to include (since it knows about at least as much information about the vertices), so it includes at least as many external edges as M'. Therefore, $|P_{12}| \geq |P'_{12}|$.

Let's calculate the utility of player 1. For every edge in P_{11}, there are two matched vertices of player 1, and for every edge in P_{12}, there is one matched vertex of Player 1. Putting it together, the utility of Player 1 under M for this path is

$$u_1(P) = 2|P_{11}| + |P_{12}| \geq 2|P'_{11}| + |P'_{12}| = u_1(P'),$$

so at least the utility for this path under M'. Hence, player 1 does not benefit from deviating in this case.

Case 2: $|P_{11}| > |P'_{11}|$. While M has more internal matches for player 1, it is possible that M' has more external edges, preventing us from arguing analogously to case 1. We need to argue that any gain in utility from additional external matches in M' is no greater than the decrease in utility due to the missing internal edges.

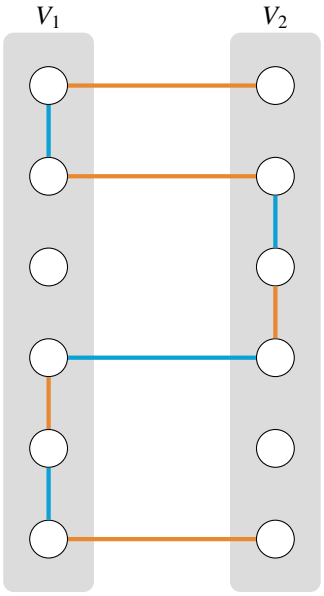
Let's consider any single subpath of the path in V_2, i.e., component of the path consisting of internal edges in V_2 between two external edges leaving V_2. If such a subpath was of odd-length, we would be able to switch the edges of one matching on the subpath with the edges of the other matching and increase the number of internal edges for player 2 in this matching. However, since both M and M' knew the same, all, vertices of player 2 and both maximized the number of internal edges for player 2, this is a contradiction. Thus, each subpath in V_2 must be of even length.

However, this tells us that the two external edges right before and right after the subpath, "entering" and "exiting" V_2, must be of opposite colors, i.e., from different matchings. Thus, we can pair up these external edges in P_{12} and P'_{12} along the path, except, potentially, the first and last external edges. Therefore, the number of external edges in P' is at most 2 more than in P, so $|P_{12}| \geq |P'_{12}| - 2$. An illustration of this argument can be found on top of the next page.

Using this, we get that

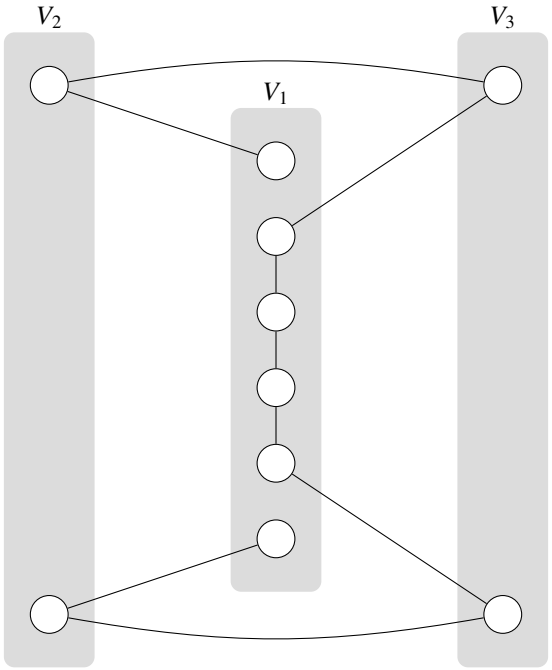
$$\begin{aligned} u_1(P) &= 2|P_{11}| + |P_{12}| \\ &\geq 2(|P'_{11}| + 1) + (|P'_{12}| - 2) \\ &= 2|P'_{11}| + |P'_{12}| \\ &= u_1(P'), \end{aligned}$$

so also in this case, player 1 cannot benefit from hiding vertices.

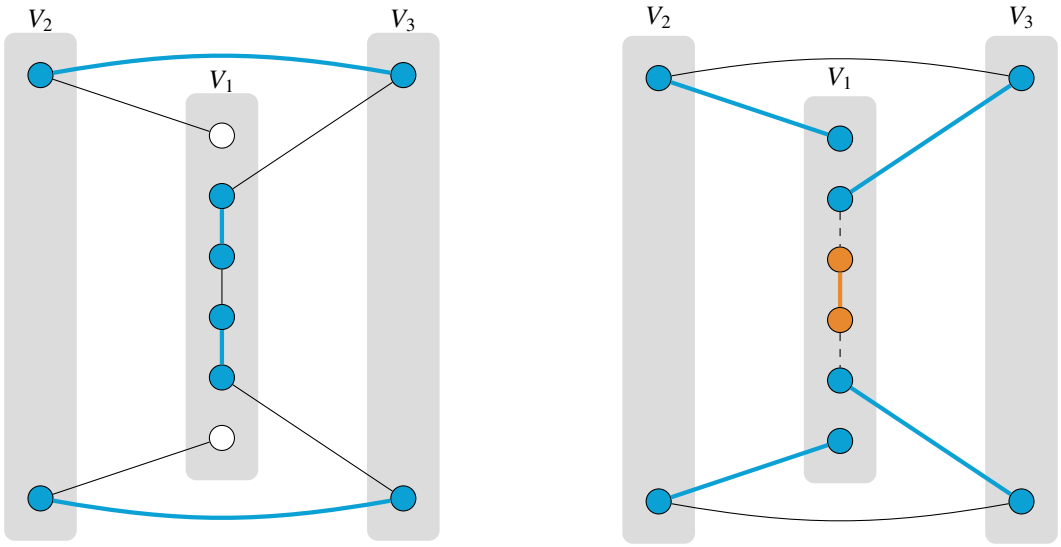


□

Unfortunately, the strategyproofness of the $\textsc{Match}_{\{\{1\},\{2\}\}}$ mechanism does not generalize to settings with more than two players. In particular, when there are three or more players, maximizing internal edges is no longer strategyproof. To see this, consider the following graph with three players.



Any mechanism that maximizes the internal edges for player 1 will result in only four vertices of player 1 being matched, as shown below on the left. However, it is possible to match all six vertices of player 1 if we are not restricted to maximizing their internal edges, as shown below. Player 1 can benefit by withholding an internal edge (orange) and matching it privately, since this results in additional external matchings that increase their utility, violating strategyproofness.



Thus, we need to search for another strategyproof mechanism.

Algorithm 2 $\textsc{Match}_{\Pi}$ Mechanism

Require: $\Pi = (\Pi_1, \Pi_2)$ is a bipartition of the set of players.

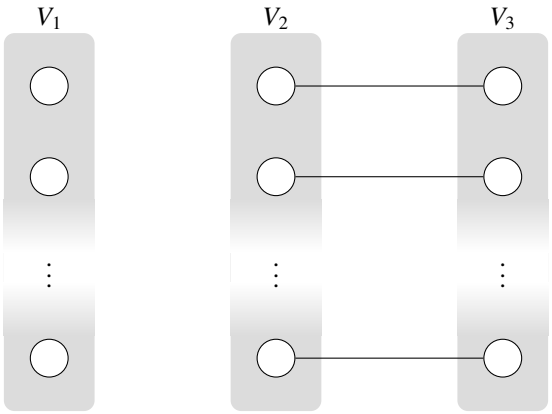
1: Consider all matchings that maximize the number of internal edges (between two vertices of the same player) for every player but contain no external edges between two players on the same side of the bipartition.
2: Among these, return the matching of maximum cardinality.

$\textsc{Match}_{\{\{1\},\{2\}\}}$ is the special case of $\textsc{Match}_{\Pi}$, with the partition Π putting one player on one side of the partition and the other player on the other side.

Theorem 3. *The* $\textsc{Match}_{\Pi}$ *mechanism is strategyproof for any number of players and any partition* Π.

The proof for this theorem is a generalization of the two-player mechanism from before.

Unfortunately, however, the $\textsc{Match}_{\Pi}$ does not have any approximation guarantee to the social welfare for three or more players. To see this, consider $n = 3$ and $\Pi = \{\{1\}, \{2, 3\}\}$ and the following graph



All edges are between players 2 and 3, so the mechanism is not allowed to match anything, since it is not allowed to match edges between players on the same side of the partition. Therefore, it will not match any patients, while in the optimal solution all vertices can be matched. The approximation ratio is 0.

However, it turns out that a randomized version of $\textsc{Match}_{\Pi}$ works surprisingly well.

Theorem 4. *The Mix-and-Match mechanism is strategyproof and gives a* $1/2$*-approximation to the optimal social welfare.*

Algorithm 3 Mix-and-Match Mechanism

1: Choose a random partition Π of the players uniformly at random.
2: Run $\textsc{Match}_\Pi$ with the chosen partition Π.

The fact that this algorithm is strategyproof follows directly from Theorem 3: Since $\textsc{Match}_\Pi$ is strategyproof for *any* partition Π, it is certainly also strategyproof for a random partition Π.

At the same time, the fact that this algorithm gives a $1/2$-approximation is intuitively very surprising in the light of the previous results, since there are two sources of loss here. First, we may be forced to take *one* internal edge instead of *two* external edges. Second, we are not allowed to match players on the same side of the partition, so every external edge is forbidden with probability $1/2$. Thus, since we are losing a factor of 2 twice, we would assume to get a $1/4$ approximation. However, it turns out these two sources of loss are mutually exclusive.