Improved Approximation for Directed Cut Problems

Amit Agarwal[*]
Dept. of Computer Science
Princeton University
35 Olden St
Princeton, NJ 08540, USA
aagarwal@cs.princeton.edu

Noga Alon[†]
Tel Aviv University
Tel Aviv 69978, Israel
and
Institute for Advanced Study
Princeton, NJ 08540, USA
nogaa@tau.ac.il

Moses Charikar[‡]
Dept. of Computer Science
Princeton University
35 Olden St
Princeton, NJ 08540, USA
moses@cs.princeton.edu

ABSTRACT

We present improved approximation algorithms for directed multicut and directed sparsest cut. The current best known approximation ratio for these problems is $O(n^{1/2})$. We obtain an $\tilde{O}(n^{11/23})$-approximation. Our algorithm works with the natural LP relaxation used in prior work. We use a randomized rounding algorithm with a more sophisticated charging scheme and analysis to obtain our improvement. This also implies a $\tilde{O}(n^{11/23})$ upper bound on the ratio between the maximum multicommodity flow and minimum multicut in directed graphs.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms

General Terms

Algorithms, Theory

Keywords

approximation algorithm, directed multicut, directed sparsest cut, linear programming relaxation

[*]Supported by NSF ITR grant CCR-0205594, NSF CAREER award CCR-0237113, MSPA-MCS award 0528414, and a Gordon Wu fellowship.

[†]Research supported in part by the Israel Science Foundation, by a USA-Israeli BSF grant, by NSF grant CCR-0324906, by a Wolfensohn fund and by the State of New Jersey.

[‡]Supported by NSF ITR grant CCR-0205594, NSF CAREER award CCR-0237113, and MSPA-MCS award 0528414,

STOC'07, June 11–13, 2007, San Diego, California, USA.

1. INTRODUCTION

Graph cut problems have a number of important applications in divide and conquer algorithms (see the survey by Shmoys [13] for details). In their seminal paper, Leighton and Rao [12] initiated the study of graph partitioning and cut problems like sparsest cut and developed basic tools for exploiting LP relaxations for these problems. A lot of progress has been made on the undirected versions of these problems starting from [12] and culminating in the recent breakthrough result of Arora, Rao and Vazirani [2] on exploiting the geometric structure of SDP relaxations for graph partitioning problems. Much less is known about the directed versions of these problems. The first non-trivial result for directed multicut was obtained by Cheriyan, Karloff and Rabani [5], who gave an LP based $O(\sqrt{n \log n})$ approximation. This was simplified and improved slightly by Gupta [7] who gave an $O(n^{1/2})$ approximation. Hajiyaghayi and Räcke [9] later obtained a matching result for the closely related problem, directed sparsest cut.

The duality between cuts and flows as well as powerful tools from metric embeddings have played an important role in the development of algorithms for undirected versions of these problems. The work of Leighton and Rao showed that the ratio of the sparsest cut to maximum multicommodity flow in undirected networks is at most logarithmic in the number of vertices. The known bounds on the corresponding flow cut ratio for directed graphs are much weaker. Low distortion embeddings of finite metrics into ℓ_1 have been valuable in the design of algorithms for undirected sparsest cut. However, directed metrics that arise in natural mathematical programming relaxations of the directed problems are much less understood than undirected metrics. Recent papers by Agarwal, Charikar, Makarychev and Makarychev [1] and Charikar, Makarychev and Makarychev [6] study directed metrics, but the results do not apply to directed multicut and directed sparsest cut that we study in this paper. Our results give $\tilde{O}(n^{11/23})$ approximations for these problems (improving on the previous best $O(n^{1/2})$) and show that there is still scope for improvement using LP based relaxations.

There is a sharp contrast between directed and undirected versions of these problems. Minimum directed multicut with two source-sink pairs is NP-hard [8], but solvable in polynomial time for the undirected version [10]. Moreover, the integrality gap of directed multicut with $k \leq \frac{\log n}{\log \log n}$ s-t pairs is $\Omega(k)$ [14], whereas for the undirected case it is

$O(\log k)$. Chuzhoy and Khanna [4] have recently made significant progress on understanding the complexity of directed multicut. They show that it is hard to approximate directed multicut and sparsest cut to within a factor of $2^{\Omega(\log^{1-\epsilon} n)}$ for any constant $\epsilon > 0$, unless $NP \subseteq ZPP$. They also show an integrality gap of $\tilde{\Omega}(n^{1/7})$ for the natural LP relaxation for this problem.

2. PRELIMINARIES

The input is a directed graph $G = (V, E)$ with positive weights $w_e : E \to R^+$ and k source-sink pairs $\{(s_i, t_i)\}_{i=1}^k$. The *minimum directed multicut* problem is to find a subset of edges C with minimum total weight $\sum_{e \in C} w_e$ that separates all the s_i, t_i pairs. An s_i, t_i pair is separated by C if there is no directed path from s_i to t_i in the graph after the removal of the edges in C. Wlog, we assume that $k \leq n^2$. In the *minimum directed sparsest cut* problem, a demand d_i is associated with the i^{th} source-sink pair. The objective is to find a subset of edges C which minimizes $\frac{\sum_{e \in C} w_e}{D_C}$ where D_C is the sum of demands of source-sink pairs separated by C. Henceforth, we focus on directed multicut. In the last section, we will explain how our results extend to the directed sparsest cut problem.

The standard LP formulation for directed multicut is as follows:

$$\min \sum_{e \in E} w_e l_e \tag{1}$$

$$s.t. \sum_{e \in p} l_e \geq 1, \quad \forall s_i - t_i \text{ paths } p, \forall i \in [k] \tag{2}$$

$$l_e \geq 0, \forall e \in E \tag{3}$$

The value of the objective for a optimum solution to the above linear program is denoted by OPT_{LP}. The LP can be solved in polynomial time using the ellipsoid algorithm with the shortest path algorithm as a separation oracle. Alternately, one can write down a polynomial sized equivalent formulation of the LP.

2.1 Definitions

For s_i-t_i, the subgraph of nodes and edges on various s_i-t_i paths is denoted by H_i. Further V_i denotes the nodes in H_i and E_i denotes the edges of H_i.

DEFINITION 2.1. *The LP volume $w(S)$, of a set of edges S, is defined to be $\sum_{e \in S} w_e l_e$.*

DEFINITION 2.2. *$d(u, v)$ = LP length of the shortest directed path from u to v induced by l_e. If v is not reachable from u, $d(u, v) = \infty$. $d_G(u, v)$ denotes the shortest path distance between u and v in graph G.*

Note that the distances in graph G are unweighted distances (i.e. they ignore weights on edges).

DEFINITION 2.3. *$diam(H) = max_{x,y \in V(H)} d_H(x, y)$ denotes the diameter of H.*

We say $u \to_i v$ iff there is a directed path from u to v in H_i.

DEFINITION 2.4. *Let $T_i(e)$ be the set of edges reachable from $e = (u, v)$ in H_i, i.e.*
$T_i(e) = \{e' = (u', v') \in E_i | \ v \to_i u'\}$.

Define $S(r, s_1) = \{v | \ d(s_1, v) \leq r\}$. This is the ball of radius r around s_1.

DEFINITION 2.5. **Level Range.** *Notation $\langle a, b \rangle_i$ will be used to denote the subgraph of H_i starting from distance a from s_i and ending at distance b from s_i. Formally,*
$\langle a, b \rangle_i \triangleq S(b, s_i) - S(a, s_i)$.

The parameter ϵ used in the algorithm and in the ensuing definitions and analysis is a small constant with value $\frac{1}{46}$.

2.2 Cuts in Level Ranges

In this section we explain how min-cuts and random level cuts are done in level ranges.

Min-Cut in $< a, b >_i$. Connect all the nodes via which paths enter from s_i into $< a, b >_i$ to a common source s and all the nodes via which paths leave $< a, b >_i$ for t_i to a common sink t by edges of infinitely large cost. Now in this graph output the minimum s-t cut.

Random Level Cut in $< a, b >_i$. A random level cut in $\langle a, b \rangle_i$ is produced as follows: Choose a number R uniformly at random in (a, b) and remove all edges $e = (u, v)$ s.t. $d(s_i, u) < R$ and $d(s_i, v) \geq R$.

2.3 Gupta's Algorithm

As a warmup we explain Gupta's algorithm and analysis [7] first:

Initially we start with an empty set of edges. The algorithm will keep adding edges to the set as it proceeds. First the algorithm solves the LP for directed multicut and includes all edges with $l_e \geq n^{-1/2}$. Then the algorithm considers s_i-t_i pairs in arbitrary order. For each s_i-t_i pair, the min-cut in $\langle 1/3, 2/3 \rangle_i$ is included in the multicut produced by the algorithm. For analysis, the cost of the min-cut in $\langle 1/3, 2/3 \rangle_i$ is charged to all the edges in H_i. We need the following lemma to formalize the cost of this cut:

LEMMA 2.6. *The cost of the min-cut in $\langle 1/3, 2/3 \rangle_i$ is at most $3w(E_i)$.*

PROOF. We use LP duality to bound the cost of the min cut in $\langle 1/3, 2/3 \rangle_i$. The LP dual is a length function on edges such that the LP length of any s-t path is at least 1. (recall that s and t are the common source and common sink introduced in producing the min cut). The cost of the min cut is then bounded above by the volume of any such LP solution. We obtain this length function by increasing the length of edges in $\langle 1/3, 2/3 \rangle_i$ by a factor of 3. Note that the LP volume of this solution is at most $3w(E_i)$ proving the lemma. ☐

Every edge in H_i receives a charge which is at most 3 times its LP contribution. In order to bound the cost of the solution, we need to bound the total charge on an edge e, i.e. bound the number of H_i's it belongs to. Edge e is said to be on the left of the min-cut if it is connected to the source after the min-cut has been produced. If e is connected to the sink then it is said to be on the right of the min-cut. We define two counters for each edge $e \in E$: $A_l(e)$ (and $A_r(e)$) which count the number of s_i-t_i pairs for which $e \in H_i$ and e occurs on the left (on the right, resp.) of the min-cut.

LEMMA 2.7. $A_l(e) + A_r(e) = O(n^{1/2})$.

PROOF. We will show that $A_l(e)$ is $O(n^{1/2})$. Let $L(e)$ be the set of indices i such that $e \in H_i$ and e lies to the left of the min-cut for s_i-t_i. Then $A_l(e) = |L(e)|$. For $i \in L(e)$, define $Q_i(e)$ to be the portion of H_i reachable from e, lying on the right of the min-cut. Note that $|Q_i(e)| \geq \sqrt{n}/3$. Further, for $i, j \in L(e), i \neq j$, we claim that $Q_i(e)$ and $Q_j(e)$ are disjoint. Suppose for contradiction, that $\exists v \in Q_i(e) \cap Q_j(e)$. Assume wlog that i came before j in the ordering. After the min-cut for s_i-t_i is produced, there is no path from e to v, contradicting the fact that $v \in Q_j(e)$. Hence $A_l(e) = |Q(e)|$ is $O(\sqrt{n})$. A similar analysis can be done for $A_r(e)$. This implies the claimed guarantee. □

The $O(n^{1/2})$ approximation follows from the previous lemma. This *disjointness argument* will play a role in our analysis as well.

3. THE MULTI-CUT ALGORITHM

Our new algorithm is the following:

Algorithm *Multi-Cut*

1. Solve the LP.
2. All edges e with $l_e \geq n^{-1/2+\epsilon}$ are added to the solution.
3. Randomly order the s-t pairs. Let the order be denoted by π.
4. **for** $j = 1$ **to** k
5. Let $i = \pi_j$. *Pick the i^{th} pair and produce a cut in the subgraph H_i as follows*:
6. Let M_i denote the min-cut in $\langle 1/3, 2/3\rangle_i$.
7. **if** $w(M_i) \leq n^{-2\epsilon} \sum_{e \in H_i} l_e w_e$.
8. **then** Add M_i to the solution.
9. **else** Add to the solution, the min-cut in $\langle 4/9, 5/9\rangle_i$ and random level cuts in $\langle 1/3, 4/9\rangle_i$ and $\langle 5/9, 2/3\rangle_i$.
10. **if** the cost of the solution returned above is more than $10000n^{1/2-\epsilon}(\log n)^{10} OPT_{LP}$
11. Output the solution returned by Gupta's algorithm;

Remark: All quantities with i in their notation are random variables that depend on the specific ordering chosen by the algorithm and the random cuts produced before the i^{th} pair.

We will deal with two kinds of distances in this paper: one is the shortest path distance according to the lengths given by the LP solution and shortest path distances in the graph itself. Since any edge that survives step 2 of the algorithm has LP length at most $n^{-1/2+\epsilon}$, a path between nodes u and v of LP length p has graph distance at least $n^{1/2-\epsilon}p$. The min-cut produced by the algorithm for some s-t pair (say s_i-t_i) divides the H_i into two parts. The side of the cut that contains the source is called the left part and the part that contains the sink is called the right part. We state the main theorem about the performance of the algorithm. The proof of the theorem is given in section 4.

THEOREM 3.1. *The expected cost of the cut returned by algorithm* **Multi-Cut** *is* $\tilde{O}(n^{11/23})OPT_{LP}$.

4. ANALYZING CHARGING SETS

4.1 Key Ideas in Our Proof

Our analysis begins with something similar to Gupta's analysis. The cost of the cuts made on line 9 of the algorithm can be bounded as follows: the cost of the min-cut is at most $9w(E_i)$. The expected cost of each random level cut is $9w(E_i)$. Therefore the total expected cost of the cuts is $27w(E_i)$. Similarly the cost of the cut added in line 8 can be bounded by $n^{-2\epsilon}w(E_i)$. We charge the cost of the cuts to edges in H_i. So the number of times an edge is charged is the same as the number of H_i's it is present in. Let's focus on an arbitrary edge e and assume that it was present in k H_i's. In such a case we can associate a special graph structure with e of size k. We formally call it a *charging set* (see definition 4.1) and denote it by P^e. Unlike Gupta's analysis, every time an edge is charged, it is typically charged $O(n^{2\epsilon})$ times its LP contribution. (details in Section 5). If $k \leq n^{1/2-3\epsilon}$, we get an improved analysis. The harder case is when k is more than $n^{1/2-3\epsilon}$ for some edges.

For each subgraph H_i that e belongs to, the size of the associated witness $|Q_i(e)|$ is $\Omega(n^{1/2-\epsilon})$. So the charging set associated with e has $kn^{1/2-\epsilon}$ nodes. For $k \geq n^{1/2-3\epsilon}$, the number of nodes is $n^{1-4\epsilon}$. Intuitively most charging sets should have many nodes in common. This is formalized by the concept of a *cover* which is a set of node-disjoint subgraphs which intersects all other charging sets in many nodes (definition 4.2).

Let us look more closely at P^e. Assume, for ease of understanding at this stage, that all nodes in P^e are present in subgraphs in C. Now one of two cases can arise:

The first case is that every subgraph H of P^e intersects few subgraphs of C (the technically precise term is *long intersections* which we define later). We redistribute the amount e was charged for H_i's it is contained in to other edges. Our analysis charges the cut for s_i-t_i only to the edges in $< 1/3, 4/9 >_i$. If $e \in < 1/3, 4/9 >_i$ is overcharged then we redistribute to $< 5/9, 2/3 >_i$. We do this for all edges $e' \in H_i$ with $\left|P^{e'}\right| \geq n^{1/2-3\epsilon}$. This is fine because we can show that the edges in $< 5/9, 2/3 >_i$ have enough LP volume to take care of such a redistribution (proved in the ratio lemma 5.2). Intuitively the LP volume should be the same everywhere in H_i because the cost of the min-cut is comparable to the total LP volume $w(E_i)$ (line 7 of the algorithm).

Next we bound the number of times edges get "re-charged" this way (denoted by $A(e)$, see definition 5.3). Here the random level cuts produced by the algorithm are useful as they decrease connectivity in subgraphs of G. The number of times the recharging happens and the number of times random level cuts are performed is the same. Using a counter based on connectivity, we bound the number of times such recharging happens (Lemma 5.9 proves this).

So far we have discussed the case when subgraphs of P^e intersect a few subgraphs of C. The second case is when they intersect with a lot of subgraphs of C. Using the main theorem and a combinatorial game over the random ordering of the s-t pairs, we can show that the total cost incurred by the algorithm in this case is $O(n^{1/2-\epsilon}OPT_{LP})$ with high probability.

4.2 Structural Definitions

In this section we define some graph structures which will help us analyze the algorithm. We associate charging sets with edges in the graph. Let $V(E)$ denote the nodes in set of edges E.

DEFINITION 4.1. *A p-charging set is a collection*

$\{G_1, \ldots, G_p\}$ of node-disjoint induced subgraphs of G s.t. $|V(G_i)| \leq n^{1/2+3\epsilon}, \forall i$. The size of a charging set is the number p of subgraphs it contains.

Let S_e be the set of s-t pairs for which e lies on the left of the min-cut. We define charging sets associated with edges e and denoted by P^e to be

$$\{V(T_i(e)) \cap \langle 5/9, 2/3\rangle_i \mid |V(T_i(e)) \cap \langle 5/9, 2/3\rangle_i| \leq n^{1/2+3\epsilon}, i \in S_e\}$$

The size of the charging set associated with an edge is connected to the number of times it is charged. We begin with some simple claims about the structure of $n^{1/2-3\epsilon}$-charging sets.

DEFINITION 4.2. *Subgraph H is said to be covered by a set of nodes N if $\forall$ path $p \in H$: $|V(p) \setminus N| \leq \frac{n^{1/2-\epsilon}}{36}$. Let S be a set of $n^{1/2-3\epsilon}$-charging sets over a set of nodes N. We say that S covers a charging set if it covers all except at most $n^{1/2-3\epsilon}$ subgraphs of the charging set.*

The reason for the value $\frac{n^{1/2-\epsilon}}{36}$ in the above definition will be clear later.

LEMMA 4.3. *$\exists$ a cover C of $n^{1/2-3\epsilon}$-charging sets, ($|C| \leq 36n^{4\epsilon}$) which covers the charging set of every edge with size more than $n^{1/2-3\epsilon}$.*

PROOF. We will construct a set C with the claimed properties. Let us start with an empty set C and consider charging sets of edges in any arbitrary order. Suppose the current charging set being considered is P^e. There are two possibilities:

1. The charging set P^e has at most $n^{1/2-3\epsilon}$ subgraphs which are not covered by $V(C)$. In this case C covers the charging set and we move on to the next charging set.

2. Otherwise, P^e has $\geq n^{1/2-3\epsilon}$ subgraphs which are not covered by $V(C)$. For each subgraph $H \in P^e$ which is not covered by C, there is at least one path which has $\frac{n^{1/2-\epsilon}}{36}$ nodes not in $V(C)$. Let one such path be denoted p_H. The new charging set is composed of such paths, one for each subgraph $H \in P^e$ not covered by C.

These paths themselves are node-disjoint and hence form a charging set. Add this charging set to C. Each charging set added increases $|V(C)|$ by at least $\frac{n^{1-4\epsilon}}{36}$. Thus the addition can happen at most $36n^{4\epsilon}$ times. □

Note that C is a function of the coin tosses made by the algorithm. Moreover, the way C is defined it is composed of charging sets which are in turn composed of paths and not subgraphs. We will abuse notation and view C as both a collection of charging sets and a collection of subgraphs/paths. The meaning intended will be clear from the context. Note that all the paths in C are node disjoint. Let C^u denote the path in C containing node u (if such a path exists).

Consider the execution of the algorithm for s_i-t_i. Let $p_e^i, \forall e \in \langle 5/9, 2/3\rangle_i$ be the probability of edge e getting cut in the random level cut in $\langle 5/9, 2/3\rangle_i$ for s_i-t_i. The reason for working with probabilities p_e^i is that the LP lengths of edges (l_e values) do not represent true probabilities of edges being cut in random level cuts. [1] Intuitively one can still think of them as values of some LP solution as they still satisfy the length constraints of the LP. Also since $p_e^i \leq 9l_e$, if a path has a probability p of being cut, it will have a graph length of at least $\frac{pn^{1/2-\epsilon}}{9}$.

DEFINITION 4.4. *For a path $p \in H_i$ and subgraph H, let $[p, H]_i$ denote the quantity $\sum_{e=(u,v)\in p: v \in V(H)} p_e^i$.*

We say that path $p \in H_i$ intersects H at long length *if*

$$[p, H]_i \geq \frac{n^{-4\epsilon}}{36 \log^3 n} \quad (4)$$

If $[p, H]_i < \frac{n^{-4\epsilon}}{36 \log^3 n}$, we say that path $p \in H_i$ intersects H at short length.

DEFINITION 4.5. *A path $p \in H_i$ is said to be formed by* short intersections *if $\exists H_1, \ldots, H_m \in C$ such that $\sum_{j=1}^{m} [p, H_j]_i \geq \frac{1}{36}$ and $\forall j \in [m], [p, H_j]_i < \frac{n^{-4\epsilon}}{36 \log^3 n}$.*

DEFINITION 4.6. *H_i is called* **bad** *if at least one of its paths is composed of short intersections.*

5. THE CHARGING SCHEME

5.1 Intuition

The charging scheme is an extension of the scheme in [7]. We first explain how the charging for the cuts in step 10 of the algorithm for H_i is done. Unlike Gupta's scheme, we cannot charge the cost to all the edges in E_i. Instead we charge to a specific subset $E'_i \subseteq E_i$ of edges with $w(E'_i) \geq \frac{n^{-2\epsilon}}{20} w(E_i)$ which we choose. The choice of E'_i is handled by lemma 5.1. The reason for doing this is the facilitation of the redistribution of charge later. Note that the algorithm **does not** need to find E'_i.

First, we show that the number of times any edge is charged due to bad subgraphs in its charging set is $n^{1/2-3\epsilon}$. If indeed some edge (say e) is charged more than $n^{1/2-3\epsilon}$ times, then charging sets in C and P^e intersect in a special way. We give a combinatorial argument to show that the contribution of these special structures to any charging set is bounded by $O(n^{1/2-3\epsilon})$ (section 5).

After this we redistribute charge for edges that have been charged more than $n^{1/2-3\epsilon}$ times. Again, some edges cannot participate in this redistribution. Lemma 5.2 shows that such edges contribute little to the LP volume overall and so do not hinder the redistribution.

Next we bound the amount of redistribution for all edges. Now we only need to deal with subgraphs in charging sets which are not bad, i.e. all their paths are formed by long intersections with subgraphs of C. Every recharge of an edge e is associated with a random level cut which destroys connectivity in the corresponding subgraph in C. The counter we use to track this is the number of ordered node pairs (x, y) s.t. y is reachable from x. Note that one can only separate a bounded number of node pairs which are at a large distance in any graph before the graph gets cut into low diameter pieces. After this happens, no large intersections are possible. This number is exactly $A(e)$ that we bound later.

5.2 Charging Lemma

In this section we show how to choose E'_i. The cost of the cuts produced in step 9 of the algorithm can be bounded

[1]Some portions of some s_i-t_i paths might be long in LP length but have small probability of being cut in a random level cut.

by $27w(E_i)$. If we can find a set E'_i such that $w(E'_i) = \Omega(n^{-2\epsilon}w(E_i))$ then the cuts in step 9 can be charged to edges in E'_i; edge $e \in E'_i$ is charged $O(n^{2\epsilon})w_e l_e$. Recall that $u \rightarrow_i v$ iff there is a directed path from u to v in H_i and $T_i(E) = \cup_{e\in E} T_i(e)$. Note once again that these quantities are defined exactly when the s_i-t_i pair was cut and are random variables which depend on the choices made by the algorithm.

LEMMA 5.1 (**Charging Lemma**). *If $w(M_i) \geq n^{-2\epsilon}w(E_i)$, then $\exists E'_i \subseteq E_i$ s.t. $w(E'_i) \geq \frac{n^{-2\epsilon}w(E_i)}{20}$ and for each $L_i \subseteq E'_i$, $w\left(T_i(L_i) \cap \langle 5/9, 2/3\rangle_i\right) \geq \frac{n^{-2\epsilon}w(L_i)}{20}$.*

PROOF. We construct such a E'_i. Initially $E'_i = E_i \cap \langle 1/3, 4/9\rangle_i$. If there is a subset $J \subseteq E'_i$ of edges for which $w\left(T_i(J) \cap \langle 5/9, 2/3\rangle_i\right) < \frac{n^{-2\epsilon}w(J)}{20}$, then $E'_i = E'_i/J$. Repeat this procedure until there is no such subset J.

Let $A_i = T_i(E_i \setminus E'_i) \cap \langle 5/9, 2/3\rangle_i$. The set E'_i that remains at the end is such that $w(A_i) < \frac{n^{-2\epsilon}w(E_i)}{20}$. The above is true as each subset J that we removed from E'_i had $w\left(T_i(J) \cap \langle 5/9, 2/3\rangle_i\right) < \frac{n^{-2\epsilon}w(J)}{20}$ and for the total set removed (i.e. $E_i \setminus E'_i$) , A_i consists of all the $T_i(J) \cap \langle 5/9, 2/3\rangle_i$.

Assume for contradiction that $w(E'_i) < \frac{n^{-2\epsilon}w(E_i)}{20}$. We will show that we can construct a cheap min-cut in H_i. Let the set of s_i-t_i paths in H_i be denoted by P_i. Construct an LP solution where

$$l'_e = 9l_e, \forall e \in A_i \cup E'_i$$

For any path $p \in P_i$, we claim that either all of its edges in $< 1/3, 4/9 >_i$ belong to E'_i or all of its edges in $< 5/9, 2/3 >_i$ belong to A_i. Suppose p contains an edge $e' \in < 1/3, 4/9 >_i$ such that $e' \notin E'_i$. Then $T_i(e') \cap < 5/9, 2/3 >_i$ is a subset of A_i. Hence all edges of p in $< 5/9, 2/3 >_i$ belong to A_i. From this, it is easy to see that the length of any path in P_i (according to length function l'_e) is at least 1.

The value of this LP solution is $9w(E'_i) + 9w(A_i) \leq \frac{18n^{-2\epsilon}w(E_i)}{20}$ which is an upper bound on the cost of the min-cut $w(M_i)$. This contradicts $w(M_i) \geq n^{-2\epsilon}w(E_i)$. Thus, $w(E'_i) \geq \frac{n^{-2\epsilon}w(E_i)}{20}$ proving the lemma. □

5.3 Ratio Lemma

Let $Q_i = P_i \cap T_i(E'_i) \cap \langle 5/9, 2/3\rangle_i$ which is the portion of the paths in P_i in $\langle 5/9, 2/3\rangle_i$ which are reachable from E'_i. For every H_i, we take another *arbitrary* subset of edges $E''_i \subseteq T_i(E'_i) \cap \langle 5/9, 2/3\rangle_i$ which satisfies the following property:

$$\forall p \in Q_i, \sum_{e\in p, e\in E''_i} p^i_e \geq \frac{1}{2}$$

We will prove that any such subset E''_i has LP volume which is comparable to the LP volume of E'_i and hence E_i.

LEMMA 5.2. [**Ratio Lemma**] *For any set E''_i of edges for which the following length constraints are true:*

$$\forall p \in Q_i, \sum_{e\in p, e\in E''_i} p^i_e \geq \frac{1}{2}$$

satisfies $w(E_i) = O\left(n^{2\epsilon}\sum_{e\in E''_i} p^i_e w_e\right)$ if step 9 of the algorithm was executed.

PROOF. To prove this we construct an LP solution (denoted by l'_e) which satisfies all the properties required of a feasible s_i-t_i length function just as in the proof of Lemma 5.1. Let $B_i = T_i(E_i \setminus E'_i) \cap \langle 5/9, 2/3\rangle_i$.

Construct an LP solution where

$$l'_e = 2p^i_e \quad \forall \quad e \in E''_i$$
$$l'_e = 9l_e, \quad \forall \quad e \in B_i$$

For any path $p \in P_i$, we claim that either it has an edge in $< 1/3, 4/9 >_i$ belong to E'_i or all of its edges in $< 5/9, 2/3 >_i$ belong to B_i. From this, it is easy to see that the length of any path in P_i (according to length function l'_e) is at least 1.

The value of this LP solution is

$$\sum_{e\in E''_i} l'_e w_e + \sum_{e\in B_i} l'_e w_e$$
$$= 2\sum_{e\in E''_i} p^i_e w_e + 9\sum_{e\in B_i} l_e w_e$$

which is an upper bound on the cost of the minimum cut $w(M_i)$. Assuming $\sum_{e\in E''_i} p^i_e w_e \leq 1/20 n^{-2\epsilon}w(E_i)$ gives a $11/40 n^{-2\epsilon}w(E_i)$ upper bound on the cost of the min-cut. This contradicts $w(M_i) \geq n^{-2\epsilon}w(E_i)$. Thus, $\sum_{e\in E''_i} p^i_e w_e \geq \frac{n^{-2\epsilon}w(E_i)}{20}$ proving the lemma. □

5.4 Analysis of Redistribution

We redistribute the charge for the set of edges $S_i \in E'_i$ which form large charging sets to edges in $T_i(S_i) \cap E''_i$ (as defined in section 4.3). We will set E''_i to be a subset of $T_i(E'_i) \cap \langle 5/9, 2/3\rangle_i$ which is formed by large intersections. Before going into the details for this case, let's take an aside into why we needed so many conditions in the earlier analysis:

We needed the special set E'_i to charge for the cost of all the cuts in H_i as for any set of edges in E'_i, we can only charge to those edges which are reachable from the set. That is why we required from E'_i that for any subset of edges L_i in it, $w\left(L_i \cap \langle 5/9, 2/3\rangle_i\right) = \Omega\left(n^{-2\epsilon}w(L_i)\right)$.

Lemma 5.2 is needed as some edges in the subgraph $T_i(E_i)$ participate in short intersections with $V(C)$ and some edges do not intersect with $V(C)$ at all. We do not bound $A(e)$ for such edges, hence avoid redistributing charge onto them.

To count the number of times edges participates in redistribution, we define a counter $A(e), \forall e = (u, v) \in E$.

Intuition for $A(e)$**.** $A(e)$ counts the number of times e will have to "handle" redistribution of charge from the set of edges S_i in H_i for different s_i-t_i pairs. For each s_i-t_i pair, $A(e)$ increases by 1 irrespective of the number of edges $e' \in E_i$ which redistribute their charge. Edge e gets charged in redistribution only if two things are true simultaneously:

1. The s_i-t_i pair has an edge e' whose charging set $P^{e'}$: $|P^{e'}| \geq n^{1/2-\epsilon}$.

2. $T_i(e') \cap \langle 5/9, 2/3\rangle_i$ has a path that uses $e = (u, v)$ at "large length".

Recall that we only redistribute charge on the portions formed by long intersections. Also note that C^v is used to denote the path containing v in the cover C.

Definition 5.3.

$$A(e) = \left|\{i | \exists e' \in E_i', \exists p \in T_i(e') \cap \langle 5/9, 2/3\rangle_i : [p, C^v]_i \geq \frac{n^{-4\epsilon}}{36 \log^3 n}\}\right|$$

We will only analyze charging sets formed by the $\langle 5/9, 2/3\rangle_i$'s for the edges that lie to the left of the min-cut.

We state the main structural theorem without proof here. It basically says that large charging sets covered by C cannot be formed by short intersections of length $\frac{n^{-4\epsilon}}{36\log^3 n}$ only.

Theorem 5.4 **(Main)**. *$\forall e \in E$, the number of bad subgraphs $H \in P^e$ is at most $n^{1/2-3\epsilon}$ with probability at least $1 - n^{-4}$.*

5.5 Cost Analysis

In this section we prove theorem 3.1 which implies the approximation factor of the algorithm. We first need the following theorem which is proved later.

Theorem 5.5. **[Re-charging Theorem].**
$\forall e, E[A(e)] = O(n^{20\epsilon} \log^7 n)$.

Now we prove theorem 3.1.

Proof. We will analyze the charge edge by edge. We will consider 4 cases which are as follows:

Case 1: The first case is for s_i-t_i pairs such that the weight of the min-cut $w(M_i) \leq w(E_i)n^{-2\epsilon}$. Let X_1 be the random variable for the cost incurred by the algorithm in this case. Also let counter $B_l(e)$ count the number of times e lies on the left of the min-cut M_i for which $w(M_i) \leq w(E_i)n^{-2\epsilon}$.

Lemma 5.6. *$B_l(e) = O\left(n^{1/2+\epsilon}\right), \forall e \in E$.*

Proof. $B_l(e) = O\left(n^{1/2+\epsilon}\right), \forall e \in E$ because of the disjointness argument which we repeat here:

Let us say some edge e lies to the left of the min-cut for two different s-t pairs, s_{i_1}-t_{i_1} and s_{i_2}-t_{i_2} and let's assume without loss of generality that i_i came before i_2 in the random order. The set of nodes reachable from e in H_{i_1} and those reachable from e in H_{i_2} and lying to the right of the min-cut are node-disjoint. Suppose for the sake of contradiction that they are not. This implies that some node to the right of the min-cut for i_1 was reachable from e even after the cut, which is a contradiction. □

In this case of there being a cheap min-cut, we charge the cost of the cut to all the edges in H_i. The charge on edge e for s_i-t_i is $n^{-2\epsilon} l_e w_e$ which implies that the total cost of such cases is bounded by $2n^{1/2-\epsilon}OPT_{LP}$ which gives

$E[X_1] = O\left(n^{1/2-\epsilon}\right) OPT_{LP}$.

Case 2: The second case is for s_i-t_i pairs such that $|H_i| \geq n^{1/2+3\epsilon}$. Let X_2 be the random variable for the cost incurred by the algorithm in this case. Also let counter $C_l(e)$ be defined just as $B_l(e)$ and denote the charge on edge e for s_i-t_i pairs for which $|H_i| \geq n^{1/2+3\epsilon}$.

Lemma 5.7. *$C_l(e) \leq O(n^{1/2-3\epsilon}), \forall e \in E$.*

Proof. Consider any edge $e \in E$, the number of H_i's such that the subgraph reachable from e has $\geq n^{1/2+3\epsilon}$ nodes is $\leq n^{1/2-3\epsilon}$ by the disjointness argument made earlier. □

We charge the cost of the cuts in this case to the set of edges E_i'. Thus each edge $e \in E_i$ gets a charge of $O\left(n^{2\epsilon} w_e l_e\right)$. Hence, the contribution of this case is also

$E[X_2] = O\left(n^{1/2-\epsilon}\right) OPT_{LP}$.

An implication of this case is that we only need to analyze charging sets composed of subgraphs of size smaller than $n^{1/2+3\epsilon}$. This will be crucial later in the proof of the re-charging theorem.

In the remaining 2 cases, we will not charge all of E_i for the cuts produced by the algorithm, but instead just the subset E_i' chosen by lemma 5.1. [2]

Case 3: In this case we bound the charge on edge e due to **bad** subgraphs and the contribution due to subgraphs that are not *covered* by C. Let X_3 be the random variable for the cost incurred by the algorithm in this case. Let counter $D_l(e)$ denote the charge on edge e for s-t pairs whose subgraphs H_i are **bad** and for which $|H_i| \leq n^{1/2+3\epsilon}$.

Lemma 5.8. *$D_l(e) \leq O(n^{1/2-3\epsilon}), \forall e$ with probability at least $1 - n^{-4}$.*

Proof. From the main theorem, with probability at least $1 - n^{-4}$ there can be no charging set of size more than $n^{1/2-3\epsilon}$ formed by short intersections. Also, at most $n^{1/2-3\epsilon}$ subgraphs in the charging set are not covered by C. If $D_l(e) > 3n^{1/2-3\epsilon}$ for some edge e, P^e is a charging set of size more than $n^{1/2-3\epsilon}$ formed by short intersections. This cannot happen with a probability of at least $1 - n^{-4}$. □

We charge the subset E_i' in this case. Note that every edge $e \in E_i'$ gets a charge of $O\left(n^{2\epsilon} w_e l_e\right)$. If $D_l(e') > 3n^{1/2-3\epsilon}$ for some edge e', the algorithm might incur a huge cost. But because of steps 10 and 11, the cost overrun is upper bounded by $n^{1/2}OPT_{LP}$.

Thus $E[X_3] = O\left(n^{2\epsilon}\left(n^{1/2-3\epsilon} + o(1)\right) OPT_{LP}\right)$

$= O\left(n^{1/2-\epsilon}\right) OPT_{LP}$.

The next case deals with re-charging the cost of edges which form large charging sets. The 4 cases handle all the possibilities.

Case 4: Let $A_l(e)$ be the number of s-t pairs for which e lies to the left of the min-cut. Note that this is exactly the same counter as defined in Gupta's algorithm. If $A_l(e) \leq 3n^{1/2-3\epsilon}$, we are fine.

Next we deal with edges e for which $A_l(e) > 3n^{1/2-3\epsilon}$. Let X_4 be the random variable for the cost incurred by the algorithm in this case. We re-distribute the charge for such edges e to edges in $< 5/9, 2/3 >_i$ for each s_i-t_i pair. There is a small complication because some edges in $< 5/9, 2/3 >_i$ might only be part of short intersections and some parts might have nodes not in common with $V(C)$. We are unable to argue anything about such portions. So we have to avoid them in the redistribution of charge. The remaining set of edges to which we can indeed re-distribute charge is denoted by E_i''. Fortunately, by lemma 5.2, $\sum_{e \in E_i''} w_e p_e^i = \Omega\left(n^{-2\epsilon} w(E_i)\right)$. Thus the amount of re-charging for one s-t

[2] Note that lemma 5.1 states that such a subset must exist unless the cost of the min-cut $w(M_i) < n^{-2\epsilon} w(E_i)$.

pair [3] on e is $O(n^{2\epsilon})p_e^i w_e$. The total expected amount of re-charging on edge $e \in E$ is $O\left(n^{2\epsilon}\left(\sum_{e\in E} E[A(e)] + n^{4\epsilon}\right)\right)$. The last additive term of $n^{4\epsilon}$ is for the charging sets in C. Therefore

$E[X_4] = O(n^{22\epsilon}\log^7 n)OPT_{LP}$ giving

E[Multi-Cut] $= O\left(n^{1/2-\epsilon} + n^{22\epsilon}poly(\log n)\right) OPT_{LP}$. The two terms are equal when $1/2 - \epsilon = 22\epsilon$. This happens for $\epsilon = \frac{1}{46}$ which gives the approximation factor claimed in the statement of theorem 3.1.

5.6 Bounding $A(e)$

Let $e = (u, v)$ and $H \in C$ be the path that contains v. The argument is based on the fact that if H contributes to $A(e)$ for a large number of edges e, then $diam(H)$ becomes small and hence no path can intersect it at large length. This implies that H cannot satisfy the conditions for increasing $A(e)$ for other edges.

Recall that P_k is the set of flow paths in H_k. Let

$$S_l(H) \quad = \quad |\{k|\ \exists p \in P_k, [p, H]_k \geq l\}|$$

(We will use a value of $\frac{n^{-4\epsilon}}{36\log^3 n}$ for l. Note that this is the same value that we use in the definition of short and large intersections.) Loosely speaking $S_l(H)$ denotes the number of s_k-t_k pairs such that H intersects some path p in H_k at large length. The main idea used in bounding $S_l(H)$ is that if H_k has a path p with a large intersection with H, two things happen at the same time:

First the probability that the random level cut for the i^{th} s-t pair lies in its intersection with H is large. Secondly, when a cut does affect H, the number of pairs of nodes in H that are separated is also large. Consider a simple case when p has LP length l common with a single path p_H of H, then the probability that the random level cut affects $p \cap H$ is at least l. Moreover when the cut occurs it is expected to separate $\Omega(n^{1-2\epsilon}l^2)$ pairs of nodes in p_H because a path of LP length l has at least $n^{1/2-\epsilon}l$ nodes. [4] This technique has also been used previously by [11].

LEMMA 5.9. $E[S_l(H)] = O\left(\frac{|H|^2\log n}{n^{1-2\epsilon}l^3}\right)$.

PROOF. Let the j^{th} s-t pair be such that $\exists p \in Q_j|\ [p, H]_j \geq \frac{n^{-4\epsilon}}{36\log^3 n}$. Let $p_1 = [p, H]_j$. The probability that the random level cuts for this pair do not effect H is at most $1-p_1 \leq e^{-p_1}$ irrespective of what happened for earlier s-t pairs.

The probability that no cut is made in H after a set S of s-t pairs with a path $p \in \langle 5/9, 2/3\rangle_j, j \in S$ which satisfies $[p, H]_j \geq \frac{n^{-4\epsilon}}{36\log^3 n}$ is at most $e^{-\sum_{i=1}^{s} Q_i}$. Hence when $\sum_{i=1}^{s} Q_i \geq 4\log n$, this probability becomes $\leq n^2 e^{-4\log n} = \frac{1}{n^2}$. Since, we can only guarantee that $Q_i \geq \frac{n^{-4\epsilon}}{36\log^3 n}, \forall i \in [1,\ldots,|S|]$, the best bound on $|S|$ we can have is $\frac{4\log n}{\frac{n^{-4\epsilon}}{36\log^3 n}}$.

Let

$$R(G) = \left|\{(x, y)|\ x, y \in V(G')\right.$$
$$\text{and there is a directed path from x to y }\}|$$

Note that $R(H) \leq |H|(|H| - 1)$.

To calculate the expected number of times cuts need to be produced in H before $diam(H)$ becomes small, we define the following random variables. Let $\Delta_k(H)$ be the reduction in $R(H)$ when the k^{th} random level cut effecting H is produced. Further let X be the random variable for the number of times cuts are produced in H_i before $diam(H)$ becomes smaller than $\frac{n^{-4\epsilon}}{36\log^3 n}$.

Intuitively if the random variables $\Delta_k(H)$ are well-behaved and each has expectation larger than M, $E[X] \leq \frac{R(H)}{M}$. To formalize the above intuition, we construct simpler random variables from the $\Delta_k(H)$'s. Let

$$\delta_k(H) = \begin{cases} 0 & \text{when } \Delta_k(H) < \frac{1}{16}n^{1-2\epsilon}\left(\frac{n^{-4\epsilon}}{36\log^3 n}\right)^2 \\ 1 & \text{when } \Delta_k(H) \geq \frac{1}{16}n^{1-2\epsilon}\left(\frac{n^{-4\epsilon}}{36\log^3 n}\right)^2 \end{cases}$$

Since any of the cuts is made at a uniformly random level, the probability that it occurs in the middle half is 1/2. When that happens, at least $\frac{1}{16}n^{1-2\epsilon}\left(\frac{n^{-4\epsilon}}{36\log^3 n}\right)^2$ u, v pairs are separated. To see why this is true, note that a LP distance of $\frac{n^{-4\epsilon}}{4x36\log^3 n}$ has at least $\frac{n^{1/2-\epsilon}n^{-4\epsilon}}{144\log^3 n}$ edges since any edge has LP length at most $n^{-1/2+\epsilon}$.

Also all such edges $ee = (u, v)$ have $v \in H$ by definition of intersections. The nodes in first one-fourth and the last one-fourth are definitely separated by the cut. Another thing to note is that the reduction in $R(H)$ of $\frac{1}{16}n^{1-2\epsilon}\left(\frac{n^{-4\epsilon}}{36\log^3 n}\right)^2$ with probability at least 1/2 is true irrespective of what happens in any of the previous cuts. This immediately gives

$$Pr\left[\delta_k(H) = 1|\ \delta_{j\neq k}(H)\right] \quad \geq \quad 1/2$$

Thus $\delta_k(H)$'s can be treated like independent Bernoulli trials. Therefore,

$$Pr\left[\sum_{1\leq k\leq m} \delta_k(H) < m/4\right] \quad \leq \quad e^{-\Omega(m)}$$

Substituting $m = \frac{4R(H)}{\frac{1}{16}n^{1-2\epsilon}l^2}$

$$Pr\left[\sum_{1\leq k\leq m} \Delta_k(H) < R(H)\right] \quad \leq \quad e^{-\Omega(m)}$$

which directly implies $Pr[X \geq m] \leq e^{-\Omega(m)} = o(\frac{1}{n^2})$ (Note that $|H| \geq 1/36n^{1/2-\epsilon}$.)

Thus $E[S_l(H)] = O\left(\frac{|H|^2}{n^{1-2\epsilon}\left(\frac{n^{-4\epsilon}}{36\log^3 n}\right)^2}\frac{\log n}{\frac{n^{-4\epsilon}}{36\log^3 n}}\right) + o(1) = O\left(\frac{|H|^2\log n}{n^{1-2\epsilon}\left(\frac{n^{-4\epsilon}}{36\log^3 n}\right)^3}\right)$. □

[3] Note that the number of times edges participate in recharging is less that the number of times they participate in random level cutting. $A(e)$ counts the number of times they can participate in random level cutting.

[4] To see this note that with probability $l/2$ the cut would be in the middle half of the intersection separating $\frac{n^{1-2\epsilon}l^2}{16}$ node pairs.

Remark: Note that we ignore the progress made due to separation of node pairs outside of $\langle 5/9, 2/3\rangle_i$.

We believe that a stronger version of lemma 5.9 might actually be true. If so, this would establish a better bound on the approximation guarantee of our algorithm. We make the following conjecture:

CONJECTURE 1. $E[S_l(H)] = O\left(\frac{|H|\log n}{n^{1/2-\epsilon}l^2}\right)$.

Now we are ready to prove our bounds on the counters $A(e)$. The main work has already been done in the previous proof.

PROOF PROOF OF THE RE-CHARGING THEOREM 5.5. Let $e = (u, v)$ be an edges and P be a charging set that contains v. Further let $H \in P$ denote the subgraph that contains v. The expected contribution to $A(e)$ due to H is exactly $E[S_l(H)]$ which by lemma 5.9 is

$O\left(\frac{|H|^2\log n}{n^{1-2\epsilon}\left(\frac{n^{-4\epsilon}}{36\log^3 n}\right)^3}\right), \forall e \in E.$

Every edge e can belong to at most 1 charging set from C (as it can belong to at most one subgraph from any charging set due to the disjointness argument made earlier).

Hence $E[A(e)] = O\left(\frac{n^{2(1/2-\epsilon)}\log n}{n^{1-2\epsilon}\left(\frac{n^{-4\epsilon}}{36\log^3 n}\right)^3}\right)$
$= O(n^{20\epsilon}\log^{10} n), \forall e \in E.$ ☐

6. PROOF OF THE MAIN THEOREM

PROOF. Suppose $\exists e \in E$ s.t. $\exists P \subseteq P^e : |P| \geq n^{1/2-3\epsilon}$ and each subgraph $H \in P$ is bad i.e. it intersects at least $36n^{4\epsilon}\log^3 n$ distinct subgraphs from C. We remove all charging sets c from C which have less than $|P|\frac{\log^3 n}{2}$ intersections with P. We denote the remaining portion of C as C'. For the i^{th} charging set $c_i \in C'$, define a_i to be the number of intersections with P and $b_i = |c_i|$. There has to be a charging set $c_j \in C'$ s.t. $\frac{a_j}{b_j} \geq \frac{\sum_{i=1}^{|C'|} a_i}{\sum_{i=1}^{|C'|} b_i} \geq \frac{|P|36n^{4\epsilon}\log^3 n}{2}/36n^{1/2+3\epsilon} \geq \frac{\log^3 n}{2}$.

We construct a zero-one matrix M with one row for each subgraph in the charging set of c_j and one column for each subgraph in the charging set P with $M_{i,k} = 1$ iff the i^{th} subgraph of c_j intersects the k^{th} subgraph of P. We delete any rows and column of M which has $\leq \frac{\log^3 n}{6}$ 1's. Suppose r rows and c columns were removed during this process, then the number of 1 entries removed from M is $\leq (r+c)\frac{\log^3 n}{6}$.

The total number of 1 entries to begin with was more than $\max\{|P|, |c_j|\}\frac{\log^3 n}{2}$. The sub-matrix M' that survives has non-trivial number of rows and columns and we denote the subsets of P, c_j that survive as P', c'_j respectively. Further the subgraphs representing the rows and columns that survive have the property that each has at least $\frac{\log^3 n}{6}$ intersections with subgraphs of the other.

Each subgraph in $c'_j \cup P'$ is part of the flow graph for some s-t pair. Let H_l denote the subgraph whose s-t pair was cut at the very end from $c'_j \cup P'$. Wlog we assume that H_l belongs to the charging set P. In case H_l belongs to c_j, we will re-define H_l to be the subgraph containing the path H_l actually represents. This small modification does not affect anything else. Further let S_1 be the set of s-t pairs from c_j that intersect H_l. We first explain a combinatorial game which abstracts out the interplay between the random ordering of s-t pairs and intersections of subgraphs in c'_j with H_l.

Let $G = (V, E)$ be a digraph with n vertices including two special ones s and t, and suppose each vertex v has an integral (time dependent) weight $w_t(v)$, where initially $w_0(v) = n^2$ for each v. We may and will assume that each vertex lies on some directed path from s to t, and hence these vertices will not be mentioned in what follows. Consider the following game played on G. In each step, an adversary is allowed to decrease the weights of any subset of the vertices arbitrarily, keeping them non-negative integers. After that, a random vertex v is chosen according to the weights, and then the weights of v and of any vertex u from which there is a directed path to v are set to 0. The game ends when all weights are 0. For a directed path P, let $g(P)$ denote the total number of vertices on P that have been chosen by the random process during the game.

Applying lemma 6.1 to the subgraph H_l and the set S_1, $|S_1| \leq \frac{\log^3 n}{6}$ with probability at least $1 - n^{-10}$ (setting $c_2 = 10$ in lemma 6.1). Next we see how the combinatorial abstracts out the situation in our case:

The weight on node $v \in H_l$ is the number of s_k-t_k pairs s.t. $v \in H_k$ and $e \in H_k$ and $e \rightarrow_k v$. These numbers can change arbitrarily depending on what cuts are produced by the algorithm in H_k, but they can only decrease. When a particular pair (say j) is chosen by the algorithm, s.t. e was on the left of the min-cut for j, we know that after the cut there is no path from e to $\langle 5/9, 2/3\rangle_j$. This can happen only if e is separated from all such H_i's from which H_j can be reached using paths in H_l. Another possibility is that the cuts are such that e is still connected to these H_i's, but the paths from them to H_j are cut. But these would include the paths in H_l as well. This immediately gives the assumption in the game that the weight of nodes from which the selected node v can be reached are reduced to 0.

Finally our condition implies that the value of the combinatorial game is larger than $\frac{\log^3 n}{6}$. We have n^6 choices as the number of possible values of the s-t pair to which H_l belongs is n^2 and the number of possible choices of charging sets P and c_j is at most n^4. This completes the proof of the theorem.

☐

6.1 A Game over Directed Graphs

Our main result about the combinatorial game is the following.

LEMMA 6.1. *For every $c_1 > 0$ there is a $c_2 > 0$ so that with probability at least $1 - \frac{1}{n^{c_1}}$, for every directed path P, $g(P) \leq c_2 \log^2 n$.*

PROOF. It is easy to prove that for any fixed path P the probability that $g(P)$ exceeds $c_2 \log^2 n$ is very small. The trouble is that there may well be an exponential number of paths, hence a first moment argument does not suffice. The idea in the proof is to define a polynomial number of potential "bad" events, show that with high probability none of them holds, and then prove that if indeed none of these bad events holds, then (deterministically), for every path P, $g(P)$ is small. Throughout the proof we have several absolute constants c_2, c_3, c_4 etc. All of those can be chosen easily as functions of the initial constant c_1 that appears in the statement of the lemma, but to simplify the presentation we do not compute them explicitly. All logarithms are in base 2.

It is convenient to describe the game in the following equivalent way. Let $V = \{y_1, y_2, \ldots, y_n\}$ be the set of vertices of G. The game will proceed in time steps, where initially the time is $t = 0$, and in time t the weight of vertex y is denoted by $w_t(y)$. Thus initially $w_0(y) = n^2$, and the adversary is allowed to decrease the weights in each step before the corresponding random choice is being performed. The random choice is a uniform, random integer x in $[1, n^3]$. If there exists an i such that $(i-1)n^2 < x \le (i-1)n^2 + w_t(y_i)$, then this corresponds to picking y_i. In this case we say that y_i is **chosen**, and its weight as well as those of all vertices u from which there is a directed path to y_i are reduced to zero. Else, we do nothing, pick another random x in the above range, and proceed as before. Obviously, this game is equivalent to the original one, where each step in the original game takes in this version a random number of steps, until some vertex is chosen. The equivalent description is convenient, as it enables us to refer to the time t during the game in a useful way. For each vertex v, let $S(v)$ denote the set of all vertices u so that there is a directed path from v to u in G, and let $w_t(S(v))$ denote the total weight of all vertices in $S(v)$ at time t, that is $w_t(S(v)) = \sum_{u \in S(v)} w_t(u)$.

Consider the following events; these are all undesirable events and we will show that each of them holds with probability at most $1/n^{c_3}$ and hence with high probability none of them holds

(i) Let E denote the event that by time $c_4 n^4$ the game still did not end.

(ii) For each integer $j \in [0, 3\log n]$, for each $t_0 \in [0, c_4 n^4]$ and for each $v \in V$, let A_{j,v,t_0} denote the event that for every time $t \in [t_0, t_0 + c_5 \frac{n^3}{2^j} \log n)$ the weight $w_t(S(v))$ is at least 2^j and also $w_t(v) > 0$.

(iii) For each j, t_0, v as above, let B_{j,v,t_0} denote the event that for every time

$$t \in I_{t_0} = [t_0, t_0 + c_5 \frac{n^3}{2^j} \log n),$$

the weight $w_t(S(v))$ is at most 2^{j+1} and during that time interval I_{t_0} there have been more than $4c_5 \log n$ choices of vertices from $S(v)$.

LEMMA 6.2. *For every $c_3 > 0$ there is a choice of $c_4, c_5 > 0$ so that the probability of E, as well as that of any single event A_{j,v,t_0} or B_{j,v,t_0} is at most $1/n^{c_3}$. Therefore, for every c_1 there are c_4, c_5 so that with probability at least $1 - 1/n^{c_1}$ none of the above events holds.*

Proof:
(i) As long as there are positive weights, the probability that at time t a vertex is chosen (and hence at least one weight of a vertex is reduced to zero) is at least $1/n^3$. Thus, the probability that after $c_4 n^4$ time steps there are still vertices with positive weights is at most the probability that the value of a binomial random variable with parameters $N = c_4 n^4$ and $p = 1/n^3$ is less than n, and this is exponentially small for any choice of a fixed $c_4 > 1$. (Note that it is in fact easy to improve the $O(n^4)$ estimate here, but this is not essential for our purpose).

(ii) As long as $w_t(S(v))$ is at least 2^j, given any history, the probability that a choice of an $x \in [1, n^3]$ will correspond to choosing a vertex in $S(v)$ (and hence reducing $w_t(v)$ to 0) is at least $2^j/n^3$. Therefore, the probability that A_{j,v,t_0} holds is at most the probability that the value of a binomial random variable with parameters $N = c_5 \frac{n^3}{2^j} \log n$ and $p = 2^j/n^3$ is 0. This is smaller than n^{-c_3} for an appropriate choice of c_5.

(iii) If $w_t(S(v))$ is at most 2^{j+1} in time t_0 (and hence stays at most 2^{j+1} during all the interval I_{t_0}), then the random variable that counts the number of choices of vertices in $S(v)$ during the interval I_{t_0} is stochastically dominated by a binomial random variable with parameters $N = c_5 \frac{n^3}{2^j} \log n$ and $p = \frac{2^{j+1}}{n^3}$. Thus, the probability that there are more than $4c_5 \log n$ such choices is smaller than $1/n^{c_3}$ for an appropriate choice of c_5. □

Returning to the proof of Lemma 6.1, observe that by the last lemma it suffices to show that if none of the events E, A_{j,v,t_0} and B_{j,v,t_0} hold, then at the end of the game, for every directed path P, $g(P) \le 12c_5 \log^2 n$. We proceed with a proof of this fact. Let $P = v_1, v_2, \ldots, v_k$ be an arbitrary directed path in G. Let $j_1 \in [0, 3\log n]$ be the integer for which $2^{j_1} \le w_0(S(v_1)) < 2^{j_1+1}$. Since none of the events $A_{j_1,v_i,t_0=0}$ holds, it follows that in time $t_1 = c_5 \frac{n^3}{2^{j_1}} \log n$ we have

$$w_{t_1}(S(v_i)) < 2^{j_1} \tag{5}$$

for every vertex on the path P whose weight in time t_1 is still positive (and in fact for all other vertices in the graph with positive weights at time t_1 as well).

As the event $B_{j_1,v_1,t_0=0}$ does not hold, it follows that until time t_1 at most $4c_5 \log n$ of the vertices of $S(v_1)$ have been chosen, and in particular, at most $4c_5 \log n$ vertices of the path P have been chosen.

Let i_2 be the first vertex along the path P whose weight in time t_1 is still positive, that is, i_2 is the minimum number such that $w_{t_1}(v_{i_2}) > 0$. Let j_2 be the integer satisfying

$$2^{j_2} \le w_{t_1}(S(v_{i_2})) < 2^{j_2+1}.$$

Note that by (5), $j_2 < j_1$.

Since none of the events A_{j_2,v,t_1} holds, it follows that in time $t_2 = t_1 + c_5 \frac{n^3}{2^{j_2}} \log n$,

$$w_{t_2}(S(v_i)) < 2^{j_2}$$

for every vertex v_i whose weight in time t_2 is still positive. As B_{j_2,v_{i_2},t_1} does not hold, it follows that between time t_1 and time t_2 at most $4c_5 \log n$ vertices of $S(v_{i_2})$ have been chosen, implying that altogether at time t_2 at most $8c_5 \log n$ vertices of the path P have been chosen.

Repeating in this manner we get after $3\log n$ stages as above that until that time, call it t, at most $3\log n 4c_5 \log n$ vertices of the path have been chosen, and for the first vertex of the path, say, v_s which is still of positive weight (if there is such a vertex at all), $w_t(S(v_s)) \le 1$, meaning that there are no additional vertices of the path following v_s with positive weight. This shows that at the end $g(P) \le 12c_5 \log^2 n$, completing the proof. □ ☐

7. APPLICATION TO DIRECTED SPARSEST CUT

We give an $O(\log n)$ reduction from directed sparsest cut to directed multicut. Hence Theorem 3.1 implies an approximation ratio of $\tilde{O}(n^{11/23})$ for directed sparsest cut. There is a well known $O(\log D)$ reduction between sparsest cut and multicut (where D is the sum of the demands) (Section 5.3.2 of [13]). We show how to use this recursively to get an $O(\log n)$ factor reduction.

Start by sorting the demands. Let the sorted list be $d_1, \ldots d_k$. Next construct k multicut problems, by assuming that d_i for each i was the largest demand separated. For demand value d_i, look at the set S_i of s-t pairs with demands in $\frac{d_i}{n^2} \leq d_{s-t} \leq d_i$. The demands are reduced to lie in $[1, n^2]$ by dividing each one of them by $\frac{d_i}{n^2}$. Applying the $O(\log D)$ reduction to S_i, we get an $O(\log n)$ reduction. Adding back the demands that were smaller than $\frac{d_i}{n^2}$ can only increase the total demand separated by a factor of 2. So our reduction is indeed a $O(\log n)$ reduction assuming d_i was the largest demand separated. Such a reduction is performed for each demand value d_i and the best output is chosen.

8. ACKNOWLEDGEMENTS

The first author would like to thank Eden Chlamtac for early discussion.

9. REFERENCES

[1] Amit Agarwal, Moses Charikar, Konstantin Makarychev, and Yury Makarychev. $O(\sqrt{\log n})$ Approximation Algorithms for Min Uncut, Min 2cnf Deletion, and Directed Cut Problems. In *STOC*, pages 573–581, 2005.

[2] Sanjeev Arora, Satish Rao, and Umesh V. Vazirani. Expander Flows, Geometric Embeddings and Graph Partitioning. In *STOC*, pages 222–231, 2004.

[3] Julia Chuzhoy and Sanjeev Khanna. Hardness of Cut Problems in Directed Graphs. In *STOC*, pages 527-536, 2006.

[4] Julia Chuzhoy and Sanjeev Khanna. Polynomial Flow-Cut Gaps and Hardness of Directed Cut Problems. To appear in *STOC* 2007.

[5] Joseph Cheriyan, Howard J. Karloff, and Yuval Rabani. Approximating Directed Multicuts. In *FOCS*, pages 320–328, 2001.

[6] Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Directed Metrics and Directed Graph Partitioning Problems. *In Proc. SODA*, 2006.

[7] Anupam Gupta. Improved Results for Directed Multicut. In *SODA*, pages 454–455, 2003.

[8] Naveen Garg, Vijay V. Vazirani, and Mihalis Yannakakis. Multiway Cuts in Directed and Node Weighted Graphs. In *ICALP*, pages 487–498, 1994.

[9] MohammadTaghi Hajiaghayi and Harald Raecke. An $O(\sqrt{n})$-Approximation Algorithm for Directed Sparsest Cut. *Information Processing Letters*. 97(4):156–160, 2006.

[10] T.C. Hu. Combinatorial Algorithms. *Addison-Wesley Publishing Company Advanced Book Program*, 1982.

[11] Yana Kortsarts, Guy Kortsarz, and Zeev Nutov. Greedy Approximation Algorithm for Directed Multicuts. *Networks*, 45(4):214–217.

[12] Tom Leighton and Satish Rao. An Approximate Max-Flow Min-Cut Theorem for Uniform Multicommodity Flow Problems with Applications to Approximation Algorithms. In *FOCS*, pages 422–431, 1988.

[13] David B. Shmoys. In Approximation Algorithms for NP-Hard Problems, *PWS Publishing*, pages 192–235, 1997.

[14] Michael E. Saks, Alex Samorodnitsky, and Leonid Zosin. A Lower Bound on the Integrality Gap for Minimum Multicut in Directed Networks. *Combinatorica*, 24(3):525–530, 2004.