Dual-Fitting Approximation Algorithms for Network Connectivity Problems

Benjamin Grimmer*

Abstract

We consider the NP-complete network connectivity problem of Dual Power Assignment (DPA). This models an ad hoc networks where each node can either operate at high or low power. The goal is to produce a minimum power strongly connected network. We give a Dual-Fitting algorithm to DPA with a 3/2-approximation ratio, improving the previous best known approximation of $11/7 \approx 1.57$. Another standard network design problem is Minimum Strongly Connected Spanning Subgraph (MSCS). We propose a new problem generalizing MSCS and DPA called Star Strong Connectivity (SSC). Then we show that our Dual-Fitting approach achieves a 1.6-approximation ratio to SSC. This Dual-Fitting approach may have applications to other connectivity programs with cut-based linear programming relaxations. As a result of our approximations, we prove new upper bounds on the integrality gaps of these problems. For completeness, we present a family of instances of MSCS (and thus SSC) with integrality gap approaching 4/3.

1 Introduction

In this paper, we consider approximation algorithms for multiple graph connectivity problems. Typically, approximation algorithms need a lower bound for the optimal solution to base their solution on. Many previous works have used matchings, greedy properties, and linear program solutions as a lower bound. Most connectivity problems in graph theory can be expressed as a cut-based integer program. The linear relaxation of this program has been used in rounding algorithms for a variety of different connectivity problems [1]. We introduce a general approach for constructing using a solution to the dual of the cut-based linear program as a lower bound. We will apply this to multiple network design problems.

One well-studied area of network design focuses on the problem of assigning power levels to vertices of a graph to achieve a connectivity property. This is useful in modeling radio networks and ad hoc wireless networks. It is common in this type of problem to minimize total power consumed by the system. This class of problems take as input a directed simple graph $G = (V, E)$ and a cost function $c : E \rightarrow \mathbb{R}^+$. A solution to this problem assigns every vertex a nonnegative power, $p(v)$. We use $H(p)$ to denote the spanning subgraph of $G$ created by a power assignment $p$ (we will formally define $H(p)$ later). The minimization

*bg@hawk.iit.edu; Illinois Institute of Technology, Chicago IL
problem then is to find the minimum power assignment, $\sum p(v)$, subject to $H(p)$ satisfying a specific property.

The first work on Power Assignment was done by Chen and Huang [2], which assumed that $E$ is bidirected (i.e. if $uv \in E$ then $vu \in E$ and $c(uv) = c(vu)$). There has been a large amount of interest in this type of problem since 2000 (some of the earlier papers are \[3, 4, 5\]). While we consider problems seeking strong connectivity, other works have focused on designing fault-tolerant networks. In \[6\], approximations for both problems seeking biconnectivity and edge-biconnectivity are given. Further, \[7\] considers the more general problems of $k$-connectivity and $k$-edge-connectivity.

We consider an asymmetric version of Power Assignment, which was shown to be NP-Complete in \[8\]. The power assignment induces a simple directed subgraph $H(p)$ on vertex set $V$ where $xy \in E(H(p))$ if and only if the arc $xy \in E$ and $p(x) \geq c(xy)$. The goal is to minimize the total power subject to $H(p)$ being strongly connected. This problem has had many different approximations proposed, which are compared in \[9\]. If we assume the input graph and cost function are bidirected, the best known approximation algorithm achieves 1.85-approximation ratio \[10\].

We are particularly interested in a special case of asymmetric Power Assignment called **Dual Power Assignment** (DPA). This problem takes a bidirected instance of Asymmetric Power Assignment with cost function $c : E \to \{0, 1\}$. This models a network where each node can operate at high or low power, and finds a minimum sized set of nodes to assign high power to produce a strongly connected network. The best known approximation for DPA was proposed by \[11\] and achieves an $11/7$-approximation. This algorithm uses interesting Hamiltonian cycle properties. C˘ alinescu in \[12\] showed that a modification of the $1.61 + \epsilon$ algorithm of Khuller et al. for **Minimum Strongly Connected Spanning Subgraph** in \[13, 14\] can be used on DPA. An algorithm for DPA with a $5/3$-approximation ratio and nearly linear runtime was given in \[15\].

### 1.1 Related Problems

Another variation of Power Assignment is Symmetric Power Assignment, which was shown to be NP-Complete in \[16\]. The power assignment induces a simple undirected graph $H(p)$ on vertex set $V$ given by $\{x, y\} \in E(H(p))$ if and only if the arc $xy \in E$ and $p(x) \geq c(xy)$ and $p(y) \geq c(xy)$. Then the goal is to minimize the total power subject to $H(p)$ being connected. As with the asymmetric problem, we can consider a subproblem with two power levels. We call this problem **Symmetric Dual Power Assignment** (SDPA). The best known approximation for SDPA achieves a $3/2$-approximation ratio \[17\]. Even if the constraint to only two power levels is removed, there is still a $3/2 + \epsilon$-approximation \[18\]. Further, the nearly linear algorithm in \[15\] also gives a $5/3$-approximation to SDPA.

One fundamental directed graph connectivity problems is **Minimum Strongly Connected Spanning Subgraph** (MSCS). This problem takes as input a strongly connected digraph and outputs a strongly connected spanning subgraph with minimum cardinality arc set. In \[19\], Vetta proposes the best known approximation for MSCS using a matching lower bound to get an approximation ratio of $3/2$. There are two other notable approximation algorithms for MSCS. First, in \[13, 14\], Khuller, Raghavachari and Young gave a greedy algorithm with a $1.61 + \epsilon$ approximation ratio. Second, in \[20\], Zhao, Nagamochi and Ibaraki
give an algorithm that runs in linear time with a 5/3-approximation ratio. This algorithm implicitly the dual linear program as a bound to get its 5/3 ratio.

When MSCS is generalized to have weights on each arc the best known algorithm is a 2-approximation. This very straightforward algorithm works by computing an in-arborescences and an out-arborescences with the same root in the digraph, and outputting their union. This is the best known algorithm even when arc weights are restricted to be in \{0, 1\}.

1.2 Our Results

One may note that MSCS and DPA both have approximation algorithms based on very similar ideas. This raises the question of how these two problems are related. To answer this question, we propose a new connectivity problem generalizing both of them called **STAR STRONG CONNECTIVITY** (SSC), defined as follows: We call a set of arcs sharing a source endpoint a **star**. SSC takes a strongly connected digraph \(G = (V, E)\) and a set \(C\) of stars as input such that \(\bigcup_{F \in C} F = E\). Then SSC finds a minimum cardinality set \(R \subseteq C\) such that \((V, \bigcup_{F \in R} F)\) is strongly connected.

SSC is exactly MSCS when all \(F \in C\) are restricted to have \(|F| = 1\). Under this restriction, choosing any star in SSC would be equivalent to choosing its single arc in MSCS. Further, we make the following claim relating SSC and DPA (proof of this claim is deferred to the appendix).

**Claim 1.** When **STAR STRONG CONNECTIVITY** has a bidirected input digraph \(G\) (an arc \(uv\) exists if and only if the arc \(vu\) exists), it is equivalent to **DUAL POWER ASSIGNMENT**.

In some sense, SSC has a more elegant formulation than DPA. It removes the complexity of having two different classes of arcs. This benefit becomes very clear when constructing Integer Linear Programs for the two problems. Besides this difference in elegance, both the resulting linear programs for DPA and bidirected SSC are equivalent to each other.

We introduce a novel Dual-Fitting approach, resulting in new approximations for these connectivity problems. This methodology utilizes the cut-based linear programming relaxation of a connectivity problem. Applying the Dual-Fitting method to DPA gives a 3/2-approximation. This improves the previous best known approximation for DPA of \(11/7 \approx 1.57\) [11]. Our algorithm and its analysis are made simpler by viewing it as a bidirected instance of SSC instead of DPA. In Section 3, we present our algorithm, prove its approximation ratio and show this bound is tight. Simple runtime arguments can show this algorithm terminates in \(O(|V||E|)\).

**Theorem 1.** **DUAL POWER ASSIGNMENT** has a Dual-Fitting 1.5-approximation algorithm.

Integer Linear Programs are often used to formulate NP-Complete problems. An approximation that uses a linear programming relaxation typically cannot give a better approximation ratio that the ratio between solutions of the integer and relaxed programs. This ratio is known as the integrality gap of a program. For minimization problems, it is formally defined as the supremum of the ratio between the optimal integer solution and the optimal fractional solution over all problem instances. As a result of our analysis for Theorem 1, we prove an upper bound of 1.5 for DPA’s integrality gap. This improves the previous upper bound of 1.85 proven in [10].
Corollary 1. The integrality gap of the standard cut-based linear program for Dual Power Assignment is at most 1.5.

Now, we turn our focus to the more general problem of SSC. The 3/2-approximation for MSCS in [19] and 11/7-approximation in [11] do not seem to generalize to SSC. However, the greedy approach used by Khuller et al. in [21] on MSCS and by Călinescu in [12] on DPA appears to generalize easily to SSC.

Claim 2. SSC has a $1.61 + \epsilon$ polynomial approximation scheme using the novel method of [13].

In Section 4, we improve on this $1.61 + \epsilon$-approximation by applying our Dual-Fitting approach to SSC. This yields an algorithm with a tight 1.6 approximation ratio. Since MSCS is a subproblem of SSC, this approximation ratio also extends to it. Simple runtime arguments can show this algorithm will terminate in $O(|V||E|)$ and $O(|V|^2|E| + |C|)$ for MSCS and SSC, respectively (where $|C| = \sum_{F \in C} |F|$). As with our approximation of DPA, we observe that an upper bound on the integrality gap follows from our analysis.

Theorem 2. Star Strong Connectivity (and thus MSCS) has a Dual-Fitting 1.6-approximation algorithm.

Corollary 2. The integrality gap of the standard cut-based linear program for Star Strong Connectivity (and thus MSCS) is at most 1.6.

In Section 4, we improve on this 1.61 + \epsilon-approximation by applying our Dual-Fitting approach to SSC. This yields an algorithm with a tight 1.6 approximation ratio. Since MSCS is a subproblem of SSC, this approximation ratio also extends to it. Simple runtime arguments can show this algorithm will terminate in $O(|V||E|)$ and $O(|V|^2|E| + |C|)$ for MSCS and SSC, respectively (where $|C| = \sum_{F \in C} |F|$). As with our approximation of DPA, we observe that an upper bound on the integrality gap follows from our analysis.

Theorem 2. Star Strong Connectivity (and thus MSCS) has a Dual-Fitting 1.6-approximation algorithm.

Corollary 2. The integrality gap of the standard cut-based linear program for Star Strong Connectivity (and thus MSCS) is at most 1.6.

A lower bound on the integrality gap of a problem provides a bound on the quality of approximation that can be achieved with certain methods. Linear program rounding, Primal-Dual algorithms, and Dual-Fitting algorithms are all limited by this value. As far as we are aware, no lower bounds on the integrality gap of MSCS exist in the literature. In Section 5, we present a family of instances of MSCS with integrality gap approaching 4/3. Since SSC is a generalization of MSCS, this bound extends to it.

Theorem 3. The integrality gap of MSCS (and thus SSC) is at least 4/3.

2 Preliminaries

The basic idea of our Dual-Fitting method is to build a feasible dual solution while constructing our integer primal solution. We construct our primal solution by finding and contracting a problem specific type of subgraph (a perfect set for SSC, defined later). Using a cut-based linear program, a dual solution will be a set of disjoint cuts (where the exact definition of disjoint is problem specific). We are interested in cuts that are disjoint from all cuts after contracting a subgraph. Later, we formally define these as internal cuts. We choose the subgraph to contract based on it having a number of disjoint internal cuts. When our algorithm terminates, the union of these disjoint cuts in each iteration will give a feasible dual solution.

Now we will construct a cut-based program for SSC and formalize all of these definitions for our algorithms. Note that we will give our approximation for DPA in terms of SSC for simplicity. For the remainder of this Section, we consider an instance of SSC on a digraph $G = (V, E)$ and a set of stars $C$. 

Definition 1. For any star $F \in C$, we define $source(F)$ to be the common source vertex of all arcs in $F$. We define $sink(F)$ to be the set of endpoints of arcs in $F$.

Definition 2. For a cut, $\emptyset \subset S \subset V$, we define $\delta(S)$ to be the set of all $F \in C$ such that $source(F) \in S$ and at least one element of $sink(F)$ is in $V \setminus S$.

This notation allows us to use $\delta(S)$ as the set of all stars with an arc crossing from $S$ to $V \setminus S$. Using these definitions, we can create a cut-based linear programming relaxation for SSC.

**SSC Primal LP**

\[
\begin{align*}
\text{minimize} & \quad \sum_{F \in C} x_F \\
\text{subject to} & \quad \sum_{F \in \delta(S)} x_F \geq 1, \quad \forall \emptyset \subset S \subset V \\
& \quad x_F \geq 0, \quad \forall F \in C
\end{align*}
\]

**Claim 3.** When SSC Primal LP is restricted to $x_F \in \mathbb{Z}$, it is exactly SSC.

**Proof.** Deferred to the Appendix. \qed

Intuitively, the dual of SSC is to find the maximum set of cuts, such that no star $F \in C$ crosses multiple of our cuts. Properly, we can consider fractional cuts in our dual problem, but our algorithm only uses integer solutions to the dual problem.

**SSC Dual LP**

\[
\begin{align*}
\text{maximize} & \quad \sum_{\emptyset \subset S \subset V} y_S \\
\text{subject to} & \quad \sum_{F \in \delta(S)} y_S \leq 1, \quad \forall F \in C \\
& \quad y_S \geq 0, \quad \forall \emptyset \subset S \subset V
\end{align*}
\]

In many previous approximations for MSCS, one repeatedly finds cycles in the digraph and contracts them. A cycle of length $k$, adds $k$ to the cost of the solution and reduces the number of vertices by $k - 1$. In [12], Călinescu proposed a novel way to extend this approach to DPA. Following from those definitions, we will use the following two definitions to define a contractible structure in SSC.

**Definition 3.** A set $Q \subseteq C$ is quasiperfect if and only if all $F \in Q$ have a distinct $source(F)$ and the subgraph with vertex set the sources of the stars of $Q$ and arc set $\bigcup_{F \in Q} F$ is strongly connected.
We will use $\text{source}(Q)$ for a quasiperfect $Q$ to be the set of all source vertices in $Q$. The distinction between $\text{source}$ defined on $F \in C$ and $\text{source}$ defined on quasiperfect sets will always be clear from context.

**Definition 4.** A set $Q \subseteq C$ is **perfect** if and only if $Q$ is quasiperfect and all $F \in Q$ have $\text{sink}(F) \subseteq \text{source}(Q)$.

We define contracting a perfect set as follows: replace all the source vertices of the perfect set with a single supervertex whose arc set is all arcs with exactly one end point in our perfect set. As a result of contraction, the size of a star may decrease, and a star may be removed if it has no remaining arcs. We can combine duplicate arcs into a single arc during this contraction process. A quasiperfect $Q$ adds $|Q|$ cost and contracts the $|Q|$ vertices of $\text{source}(Q)$, but may have extra arcs leaving the new supervertex. A perfect set has no such arcs, so the problem after contracting such a set will be another instance of SSC. Our next lemma describes how to expand any quasiperfect set into a perfect set.

**Lemma 1.** Every quasiperfect set is a subset of some perfect set.

*Proof.* Consider the following expansion procedure for some quasiperfect set $Q$.

1. **while** $\exists F \in Q$ with $u \in \text{sink}(F) \setminus \text{source}(Q)$ **do**
2. Find a path $p$ from $u$ to $\text{source}(Q)$ that is internally disjoint from $Q$
3. **for each** arc $e$ in $p$ **do** add a star containing $e$ to $Q$ **end for**
4. **end while**

Any star added must have had source outside of $\text{source}(Q)$. So no star added will share a source vertex with any other star in $Q$. Further $Q$ will still have a strongly connected subgraph. Thus each iteration of this procedure maintains the invariant that $Q$ is quasiperfect. When this construction terminates, no such $F$ exists. Therefore $Q$ is perfect. Each iteration of this procedure increases the size of $Q$, so it will terminate eventually. Line 2 of this construction can be implemented using a simple depth first search.

Our expansion procedure can be simplified slightly for DPA. Since the graph is bidirected, lines 2 and 3 can just choose any star containing the reverse arc from $u$ to $\text{source}(F)$. For the special case of MSCS, all stars of size one. It follows that all quasiperfect sets will be perfect. In fact, for MSCS, it can easily be shown that all quasiperfect and perfect sets are cycles.

Our approximation algorithms will repeatedly find perfect sets and contract them. The dual problem requires us to build a set of cuts that share no crossing stars. We use the following two definitions to formalize this for SSC:

**Definition 5.** Two cuts $S_1$ and $S_2$ are **disjoint** if and only if $\delta(S_1)$ and $\delta(S_2)$ are disjoint (i.e. $\delta(S_1) \cap \delta(S_2) = \emptyset$).

Note that if two cuts share no vertices, then they also also share no crossing stars (i.e. they are disjoint).

**Definition 6.** A cut $S$ is **internal** to a set $Q \subseteq C$ if and only if every $F \in \delta(S)$ has $\text{source}(F) \in \text{source}(Q)$ and $\text{sink}(F) \subseteq \text{source}(Q)$. 
3 1.5-Approximation for DPA

In Lemma 2, we give a construction for a perfect set with two disjoint internal cuts. Utilizing this Lemma, our approximation algorithm becomes very simple. Our algorithm will repeatedly apply this construction and contract the resulting perfect set. This procedure is formally given in Algorithm 1.

Algorithm 1 Dual-Fitting Algorithm for Dual Power Assignment

1: \( R = \emptyset \)
2: \( \textbf{while} |V| \neq 1 \textbf{ do} \)
3: \( \text{Find a perfect set } Q \text{ with two disjoint internal cuts as shown in Lemma 2} \)
4: \( \text{Contract the sources of } Q \text{ into a single vertex} \)
5: \( R := R \cup Q \)
6: \( \textbf{end while} \)

Lemma 2. Every bidirected instance of SSC has a perfect set with two disjoint internal cuts.

Proof. We consider an instance of SSC defined on a bidirected digraph \( G = (V, E) \). In the degenerate case, we have a graph with only two vertices, \( u \) and \( v \). Then the perfect set \( \{uv\}, \{vu\} \) will have disjoint internal cuts \( \{u\} \) and \( \{v\} \). Now we assume \( |V| \geq 3 \). Let \( n(v) \) denote the neighbors of a vertex \( v \). Note that the set of in-neighbors and out-neighbors for a vertex are identical since the graph is bidirected. We call any vertex with exactly one neighbor a leaf. Consider the following cycle construction. Examples of its output are given in Figure 1.

1: Set \( p \) to be any arc \( uv \in E \) where \( v \) is not a leaf
2: \( \textbf{while} \) TRUE \( \textbf{ do} \)
3: \( \text{if } \exists u \in V \text{ s.t. } u \in n(\bar{v}) \setminus V(p) \text{ and } u \text{ is not a leaf } \textbf{ then} \)
4: \( p := p \text{ concatenated with the arc } \bar{v}u \)
5: \( \bar{v} := u \)
6: \( \textbf{else} \)
7: \( \text{Set } w \text{ to be the vertex in } n(\bar{v}) \cap V(p) \text{ earliest in } p \)
8: \( \text{Set } \bar{w} \text{ to be the successor of } w \text{ in } p \)
9: \( \textbf{if } \exists u \in V \text{ s.t. } u \in n(\bar{w}) \setminus V(p) \text{ and } u \text{ is not a leaf } \textbf{ then} \)
10: \( \text{Replace } p \text{ with the path using } w\bar{v} \text{ instead of } w\bar{w}, \text{ reversing all arcs between } \bar{v} \text{ and } \bar{w} \)
11: \( p := p \text{ concatenated with the arc } \bar{w}u \)
12: \( \bar{v} := u \)
13: \( \textbf{else} \)
14: \( \text{Set } x \text{ to be the vertex in } n(\bar{w}) \cap V(p) \text{ earliest in } p \)
15: \( \text{Set } C \text{ to the cycle using the arc } \bar{w}x, \text{ the reverse of arcs in } p \text{ from } w \text{ to } x, \text{ the arc } x\bar{w}, \text{ and arcs in } p \text{ from } \bar{w} \text{ to } \bar{v} \)
16: \( \textbf{return} C, \bar{v}, \bar{w} \)
17: \( \textbf{end if} \)
18: \( \textbf{end if} \)
Figure 1: Examples of cycles produced by our construction for Lemma 2. Dashed curves represent a path. We do not show the leaves that may exist next to $\bar{v}$ or $\bar{w}$. (a) Shows the general form of our cycle. (b) Shows the special case when $|C| = 2$.

19: end while

**Claim 4.** For any bidirected instance of SSC with $|V| \geq 3$, this construction will output a cycle $C$ with vertices $\bar{v}, \bar{w} \in V(C)$ having the following two properties:

- $\bar{v}$ and $\bar{w}$ are not leaves
- Each neighbor of $\bar{v}$ or $\bar{w}$ is either in $V(C)$ or a leaf.

**Proof.** Any graph with at least three vertices will have an initial edge $u\bar{v}$ where $\bar{v}$ is not a leaf. This guarantees that step 1 is possible. Then each iteration increases the length of the path $p$. It follows that there are at most $|V|$ iterations before the construction terminates.

For our first property, we maintain the invariant that $\bar{v}$ is not a leaf. This is true from our initial choice of $\bar{v}$, and also maintained in each $u \in V$ chosen to extend $p$. Finally, $\bar{w}$ is either equal to $\bar{v}$ and thus not a leaf, or inside the path $p$ and thus has two neighbors.

For our second property, the choice of the cycle $C$ implies that $V(C)$ contains all vertices in $p$ between $x$ and $\bar{v}$. All non-leaf neighbors of $\bar{w}$ are at most as early as $x$ in $p$. All non-leaf neighbors of $\bar{v}$ are at most as early as $w$ in $p$. Note that $w$ is at most as early as $x$ in $p$. Then all the non-leaf neighbors of $\bar{v}$ and $\bar{w}$ must be in the $V(C)$.

Using this cycle $C$, we will construct our perfect set with two disjoint internal cuts. Let $L_{\bar{v}}$ and $L_{\bar{w}}$ be the set of leaves adjacent to $\bar{v}$ and $\bar{w}$, respectively. Consider the case where there is a star $F$ sourced at $\bar{v}$ containing arcs to multiple leaves. Let $l_1$ and $l_2$ be two distinct leaves in $\text{sink}(F)$. Then we expand the quasiperfect set $\{F\}$ into a perfect set $Q$ using Lemma 1. This $Q$ will have internal cuts $\{l_1\}$ and $\{l_2\}$. The same construction can be made for such a star sourced at $\bar{w}$. For the remainder of our proof, we can assume no star exists sourced from $\bar{v}$ or $\bar{w}$ going to multiple leaves. Now we consider two separate cases: $\bar{v} = \bar{w}$ and $\bar{v} \neq \bar{w}$.

**Case 1:** $\bar{v} = \bar{w}$. In this case, $w$ must be the predecessor of $\bar{v}$. We can conclude that $|C| = 2$. Further, $\bar{v}$ is only adjacent to $w$ and leaves. We know that $\bar{v}$ is not a leaf. Therefore the set $L_{\bar{v}}$ must be non-empty. Let $l \in L_{\bar{v}}$ be a leaf of $\bar{v}$.
Lemma 3. The number of perfect sets of size \(i\) contracts \(i - 1\) vertices, and over the whole algorithm, we contract \(n\) vertices into 1. Therefore \(\sum_{i=2}^{n}(i-1)A_i = n - 1\). Each perfect set of size \(i\) contributes \(i\) cost to our solution. Then our cost is \(\sum_{i=2}^{n}iA_i = n + k - 1\).

**Proof.** Each perfect set of size \(i\) contracts \(i - 1\) vertices, and over the whole algorithm, we contract \(n\) vertices into 1. Therefore \(\sum_{i=2}^{n}(i-1)A_i = n - 1\). Each perfect set of size \(i\) contributes \(i\) cost to our solution. Then our cost is \(\sum_{i=2}^{n}iA_i = n + k - 1\).

Lemma 4. \(|OPT(I)| \geq n\)

**Proof.** Consider the dual solution of assigning one to the cut \(\{v\}\) for all \(v \in V\). This dual feasible solution has objective \(n\). The lemma follows from weak duality.

Lemma 5. \(|OPT(I)| \geq 2k\)

**Proof.** Whenever the algorithm adds a perfect set, we can identify two disjoint internal cuts. From the definition of internal cuts, all of these cuts will be disjoint from previously added internal cuts. So we have a dual feasible solution with objective \(2k\). The lemma follows from weak duality.
By taking a convex combination of Lemmas 4 and 5, we know the following:

\[ |OPT(I)| \geq \frac{2}{3}n + \frac{1}{3}(2k) \tag{1} \]

Taking the ratio of Lemma 3 and Equation (1), we get a bound on the approximation ratio. Straightforward algebra on this ratio completes our proof of Theorem 1:

\[ \frac{|A(I)|}{|OPT(I)|} \leq \frac{n + k - 1}{\frac{2n}{3} + \frac{2k}{3}} < \frac{3}{2} = 1.5 \]

As a corollary, 1.5 upper bounds the ratio between integer primal and integer dual solutions to our program. This is easily verified on any SSC instance by choosing \( A(I) \) for the integer primal and the larger of the two dual solutions used in Lemma 4 and 5. Corollary 1 follows from this observation.

### 3.2 Tightness of 1.5-Approximation Ratio

**Theorem 4.** The 1.5-approximation ratio of Algorithm \( I \) is tight.

**Proof.** We prove this by giving a family \( G_k \) of bidirected SSC instances where our algorithm can choose arbitrarily close to \( 3|OPT(G_k)|/2 \) stars. Our family of instances will only have stars of size one. Therefore, we can represent an instance of SSC using only the corresponding digraph. Further, since the graph must be bidirected, we can represent it using an undirected graph.

We define our family \( G_k \) as follows: Let \( V(G_k) = \{\bar{v}, \bar{w}\} \cup \{u_1, u_2, \ldots, u_{k+1}\} \cup \{l_1, l_2, \ldots, l_k\} \). Further, \( E(G_k) \) contains \( \bar{v}u_{k+1}, \bar{w}u_k, \) and all edges in the cycle \( \bar{v}, u_1, u_2, \ldots, u_{k+1}, \bar{w} \) and the cycle \( \bar{v}, u_1, l_1, u_2, l_2, \ldots, l_k, u_{k+1}, \bar{w} \). An example instance of \( G_k \) is depicted in Figure 2.

![Figure 2: Instance of \( G_k \) used to show tightness of our 1.5-approximation ratio.](image)

Suppose Algorithm \( I \) is run on \( G_k \). When our cycle construction is run, it could build the path \( u_{k+1}, \bar{w}, u_k, \ldots, u_2, u_1, \bar{v} \) before terminating. Then it would choose the perfect set corresponding to the cycle \( \bar{v}, u_{k+1}, \bar{w}, u_k, \ldots, u_2, u_1 \). This set has disjoint internal cuts \( \{\bar{v}\} \) and \( \{\bar{w}\} \). After this is contracted into a vertex \( s \), the algorithm will have to choose the perfect set of size two contracting \( l_i \) into \( s \) for each \( 1 \leq i \leq k \). Each of these sets have disjoint internal cuts \( \{l_i\} \) and \( V \setminus \{l_i\} \). Therefore our algorithm could choose \( 3k + 3 \) stars.

The optimal solution to \( G_k \) will choose the perfect set corresponding to the Hamiltonian cycle \( \bar{v}, u_1, l_1, u_2, l_2, \ldots, l_k, u_{k+1}, \bar{w} \). This solution has objective \( 2k + 3 \). Then the approximation
ratio achieved on $G_k$ could be as large as $(3k + 3)/(2k + 3)$. As $k$ approaches infinity, the ratio achieved on $G_k$ approaches $3/2$. \hfill \Box

4 1.6-Approximation for SSC

As in our approximation for DPA, we need a method to construct perfect sets with internal cuts. Without the restriction to bidirected input digraphs, we are unable the guarantee two disjoint internal cuts. Following from our analysis of DPA, such a construction would give SSC a 1.5-approximation. Instead, we guarantee the following weaker condition.

Lemma 6. Every instance of SSC has a perfect $Q$ with either $|Q| \geq 4$ and one internal cut, or two disjoint internal cuts.

Proof. We use $n^+(v)$ to denote the set of out-neighbors of a vertex $v$ in $G$. We use the following cycle construction, which is a simplification of the construction used for DPA.

1: Set $p$ to any arc in $G$
2: Set $\bar{v}$ to be the last vertex in the path $p$
3: while $\exists u \in n^+(\bar{v}) \setminus V(p)$ do
4: $p := p$ concatenated with the arc $\bar{vu}$
5: $\bar{v} := u$
6: end while
7: Set $w$ to be the vertex in $n^+(\bar{v})$ earliest in $p$
8: Set $C$ to the cycle using $\bar{vw}$ and arcs in $p$
9: return $C$ and $\bar{v}$

Claim 5. This construction will output a cycle $C$ and $\bar{v} \in V(C)$ such that $n^+(\bar{v}) \subseteq V(C)$.

This claim follows immediately from our choice of $C$. Using this $C$ and $\bar{v}$, we will construct our perfect set. We consider this in three separate cases: $|C| \geq 4$, $|C| = 3$, and $|C| = 2$.

Case 1: $|C| \geq 4$. We can construct a quasiperfect set by choosing a star containing each arc of our cycle. Then let $Q$ be the perfect set created by expanding this set using Lemma 1. Note that $|Q| \geq 4$. Since $n^+(\bar{v}) \subseteq V(C)$, the cut $\{\bar{v}\}$ will be internal to $Q$.

Case 2: $|C| = 3$. Our cycle construction must have found a cycle $C$ on vertices $\{\bar{v}, u_1, u_2\}$. Then $\bar{v}$ has the property that $adj^+(\bar{v}) \subseteq \{u_1, u_2\}$. Suppose any of the cycle arcs, $\bar{vu}_1$, $u_1u_2$, or $u_2\bar{v}$, are part of a $F \in C$ with $u_4 \in \text{sink}(F)$ where $u_4 \notin V(C)$. Then we can construct a quasiperfect set containing this star and a star for each other arc in the cycle. Expanding this quasiperfect set, as defined in Lemma 1, will produce a perfect set of size four or more with the internal cut $\{\bar{v}\}$. So we assume that no such $F$ exists.

If there exists a nontrivial path from $u_1$ to $u_2$ or from $u_2$ to $\bar{v}$ internally disjoint from $V(C)$, then we can replace an arc of $C$ with this path to get a larger cycle that has the same property with respect to $\bar{v}$ (nontrivial meaning with $|p| \geq 2$). Then we can apply Case 1 to handle the new cycle. Similarly, at least one of the following doesn’t exist: nontrivial path from $u_1$ to $\bar{v}$ internally disjoint from $V(C)$, nontrivial path from $u_2$ to $u_1$ internally disjoint from $V(C)$ or the arc from $\bar{v}$ to $u_2$. If these all existed,
we can construct a larger cycle with the same property with respect to $\bar{v}$ by starting at $\bar{v}$, following the arc to $u_2$, following the nontrivial path to $u_1$, finally following the nontrivial path to $\bar{v}$. We know the paths from $u_2$ to $u_1$ and $u_1$ to $\bar{v}$ are internally disjoint because any overlap would create a path from $u_2$ to $\bar{v}$. Thus this construction will produce a larger simple cycle. Then we can assume at least one of these three structures does not exist.

We handle the three possible cases of our assumption separately. Let $R_C(u)$ be the set of vertices reachable by $u$ without using any arcs with both endpoints in $C$.

**No nontrivial path from $u_1$ to $\bar{v}$ exists.** Note that we also assume no nontrivial path from $u_1$ to $u_2$ exists. Then $R_C(u_1)$ doesn’t include $\bar{v}$ or $u_2$. The two cuts given by $\{\bar{v}\}$ and $R_C(u_1)$ are disjoint. Further, they are both internal to $C$, since all their arcs have both endpoints in $C$.

**No nontrivial path from $u_2$ to $u_1$ exists.** Note that we also assume no nontrivial path from $u_2$ to $\bar{v}$ exists. Then $R_C(u_2)$ doesn’t include $\bar{v}$ or $u_1$. The two cuts given by $\{\bar{v}\}$ and $R_C(u_2)$ are disjoint. Further, they are both internal to $C$, since all their arcs have both endpoints in $C$.

**No arc from $\bar{v}$ to $u_2$ exists.** Consider the two cuts given by $\{\bar{v}\}$ and $\{\bar{v}\} \cup R_C(u_1)$. The former cut is internal to $C$ from the definition of $\bar{v}$. Further, we know that no nontrivial path exists from $\bar{v}$ to $u_2$ internally disjoint from $V(C)$ and our assumptions imply that no nontrivial path exists from $u_1$ to $u_2$. Then we can conclude that the cut $\{\bar{v}\} \cup R_C(u_1)$ is only crossed by arcs from $u_1$ or $\bar{v}$ to $u_2$. Then this cut must be internal to $C$. Thus both cuts are internal to $C$ and they are disjoint because the arc $\bar{v} u_2$ does not exist.

Thus under any case we can find two disjoint cuts in our three cycle. Therefore, if we fail to replace $C$ with a larger cycle, the perfect set contracting $C$ will have two disjoint internal cuts.

**Case 3: $|C| = 2$.** Our cycle construction must have found a cycle $C$ on vertices $\{\bar{v}, u_1\}$. Note that the only arc leaving $\bar{v}$ goes to $u_1$, and there is a star containing only this arc. If a nontrivial path $p$ exists from $u_1$ to $\bar{v}$, then we can replace $C$ with the cycle made by concatenating the arc $\bar{v} u_1$ with $p$. This larger cycle can then be processed by either Case 1 or 2. So we can assume that the only path from $u_1$ to $\bar{v}$ is the arc between them. Consider the cut $V \setminus \{\bar{v}\}$, which is only crossed by $u_1 \bar{v}$. Either there exists a $F_1 \in C$ containing $u_1 \bar{v}$ and some $u_1 u_2$, or this cut is internal to any perfect set contracting $\bar{v}$ and $u_1$. In the latter case, we can choose the perfect set $\{\bar{v} u_1\}, \{u_1 \bar{v}\}$. This perfect set has disjoint internal cuts $\{\bar{v}\}$ and $V \setminus \{\bar{v}\}$.

If this $F_1$ and $u_2$ exist, then any nontrivial path $p$ from $u_2$ to $u_1$ would create a quasiperfect set of size at least four. Then by Lemma 1 we could find a perfect set of size four or more with the internal cut $\{\bar{v}\}$. Finally, we handle the case where the only path from $u_2$ to $u_1$ is the arc $u_2 u_1$. Let $R \subseteq V$ be the set of all vertices that can be reached by $u_2$ without using the arc $u_2 u_1$. Consider the cut given by $R$, which is only crossed by $u_2 u_1$. Either there exists a $F_2 \in C$ containing $u_2 u_1$ and some $u_2 u_3$, or $R$ is
internal to any perfect set contracting $u_1$ and $u_2$. In the former case, expanding the quasiperfect set $\{F_1, F_2\}$, as defined in Lemma 1, will give a perfect set of size at least four with internal cut $\{\bar{v}\}$. In the latter case, expanding the quasiperfect set $\{F_1\}$, as defined in Lemma 1, will give a perfect set with two disjoint internal cuts: $\{\bar{v}\}$ and $R$.

Therefore regardless of the size of $C$, we can construct either a size four or more perfect set with an internal cut or a smaller perfect set with two disjoint internal cuts. This concludes our proof of Lemma 6. □

Using Lemma 6 our approximation algorithm is very simple. We repeatedly apply this construction and contract the resulting perfect set. This procedure is formally given in Algorithm 2.

\begin{algorithm}
\caption{Dual-Fitting Algorithm for STAR STRONG CONNECTIVITY}
\begin{algorithmic}
\STATE $R = \emptyset$
\WHILE {$|V| \neq 1$}
\STATE Find a perfect set $Q$ as shown in Lemma 6
\STATE Contract the sources of $Q$ into a single vertex
\STATE $R := R \cup Q$
\ENDWHILE
\end{algorithmic}
\end{algorithm}

\subsection{Analysis of 1.6-Approximation Ratio}

The analysis of our approximation ratio is very similar to the analysis given in Section 3.1. Let $\mathcal{I}$ be the set of all possible SSC problem instances. We denote the solution from our algorithm on some $I \in \mathcal{I}$ as $A(I)$, and the optimal solution as $OPT(I)$. We let $A_i$ denote the number of perfect sets of size $i$ added by the algorithm.

\begin{lemma}
$|A(I)| = \sum_{i=2}^{n} i A_i$
\end{lemma}

\begin{proof}
Each perfect set in $A_i$ has $i$ stars, and thus contributes $i$ cost to our solution. □
\end{proof}

\begin{lemma}
$|OPT(I)| \geq n > \sum_{i=2}^{n} (i - 1)A_i$
\end{lemma}

\begin{proof}
Consider the dual solution of assigning one to the cut $\{v\}$ for all $v \in V$. This dual feasible solution has objective $n$. Since each perfect set of size $i$ added by the algorithm contracts $i - 1$ vertices, we know that $n - 1 = \sum_{i=2}^{n} (i - 1)A_i$. The lemma follows from weak duality. □
\end{proof}

\begin{lemma}
$|OPT(I)| \geq 2A_2 + 2A_3 + \sum_{i=4}^{n} A_i$
\end{lemma}
Proof. Whenever the algorithm adds a perfect set of size two, we can identify two disjoint internal cuts. Similarly, there are two disjoint internal cuts in every size three perfect set and one internal cut in the remaining perfect sets. From the definition of internal cuts, we know that all of these internal cuts will be disjoint from previously added internal cuts. So we have a dual feasible solution and the lemma follows from weak duality.

By taking a convex combination of Lemmas 8 and 9, we know the following:

\[ |OPT(I)| > \frac{3}{4} \left( \sum_{i=2}^{n} (i-1)A_i \right) + \frac{1}{4} (2A_2 + 2A_3 + \sum_{i=4}^{n} A_i) \]

\[ = \frac{5}{4} A_2 + 2A_3 + \sum_{i=4}^{n} \left( \frac{3}{4} i - \frac{1}{2} \right) A_i \]

Taking the ratio of Lemma 7 and Equation (3), we get a bound on the approximation ratio. Straightforward algebra on this ratio can show it is at most 8/5:

\[ \frac{|A(I)|}{|OPT(I)|} < \frac{\sum_{i=2}^{n} iA_i}{\frac{5}{4} A_2 + 2A_3 + \sum_{i=4}^{n} \left( \frac{3}{4} i - \frac{1}{2} \right) A_i} \leq \frac{8}{5} = 1.6 \]

This finishes the proof of Theorem 2. Corollary 2 follows from the same observation made about our approximation for DPA.

4.2 Tightness of 1.6-Approximation Ratio

Theorem 5. The 1.6-approximation ratio of Algorithm 2 is tight.

We show our ratio is tight by giving a simple family of digraphs where our algorithm may choose arbitrarily close to a 1.6-approximation. We use an example where all \(|F| = 1\) (i.e. when SSC is equivalent to MSCS). This allows us to describe any instance uniquely by giving its digraph. Figure 3 gives examples of our family of digraphs, \(T_k\). Formally, \(T_k\) is recursively defined as follows:

First, \(T_1\) is a digraph with vertices \(\{a, b, c, d, x, y, z\}\) and arcs of the cycles \(abcdxyz\) and \(yxca\). Each \(T_k\) will have four specific vertices denoted by \(c, d, x\) and \(y\). To construct \(T_{k+1}\) from \(T_k\), we replace \(x\) with the vertices \(\{a', b', c', d', x', y'\}\). The arc from \(x\) to \(c\) is replaced with the arc from \(a'\) to \(c\). The arc from \(d\) to \(x\) is replaced with the arc from \(d \rightarrow a'\). Similarly the two arcs between \(x\) and \(y\) are replaced with two arcs between \(y'\) and \(y\). Further, we connect these new vertices with the arcs of the paths \(a'b'c'd'x'y'\) and \(y'x'c'a'\). The vertices \(c', d', x'\) and \(y'\) from our expansion of \(T_k\) are \(c, d, x\) and \(y\) for \(T_{k+1}\) respectively.

We prove the following two lemmas about \(T_k\) to establish the approximation ratio of \(T_k\) approaches 1.6 as \(k\) grows.

Lemma 10. Every \(T_k\) has a Hamiltonian cycle containing the path \(dxy\) and of length \(5k + 2\).
Figure 3: Examples of the digraphs $T_k$ used to show tightness of our approximation bound. 
(a) $T_1$ (b) $T_2$ (c) $T_5$

Proof. We prove this by induction. By the definition of $T_1$, it contains the Hamiltonian cycle $abcdxyz$. Then for our inductive step, we assume there is such a Hamiltonian cycle $C$ in the digraph $T_k$. We consider the path made by the arcs of $C$ in $T_{k+1}$ (note the arc $xy$ becomes the arc $y'y$ and $dx$ becomes $da'$). Then the arcs of $C$ form a path starting at $y'$, going through all vertices common with $T_k$ and ending at $a'$. Concatenating this with the path $a'b'c'd'x'y'$ will yield a Hamiltonian cycle in $T_{k+1}$. Note this cycle contains the path $d'x'y'$. We added five new arcs to this cycle, giving a total size of $5k + 2 + 5 = 5(k + 1) + 2$, which completes our inductive proof.

Lemma 11. Our algorithm may choose $8k + 2$ arcs on input $T_k$.

Proof. We prove this by induction. For $T_1$, our algorithm may choose to contract the cycle $yxca$ since it has internal cut $\{x\}$. Let $w$ be the resulting supervertex. The next three iterations of our algorithm will contract the cycles $wd$, $wb$ and $wz$. Total this choose 10 arcs, confirming our base case.

Now we assume our algorithm will produce a solution to $T_k$ using $8k + 2$ arcs. Given an instance of $T_{k+1}$, consider the six vertices added in our recursive construction, $\{a', b', c', d', x', y'\}$. As in our base case, the algorithm may contract the cycle $y'x'c'a'$ into a supervertex $w$. Then it can contract the cycles $wd'$ and $wb'$. After these contractions, the six vertices that replaced $x$ in $T_k$ have been combined to a single vertex. Then it follows that after our algorithm selects these 8 arcs and contracts, $T_{k+1}$ becomes an instance of $T_k$. By our inductive assumption, this process could choose $8 + (8k + 2) = 8(k + 1) + 2$ arcs.

From Lemma [10] we know that the optimal solution to $T_k$ costs $5k + 2$. Combining this result with Lemma [11] we find $T_k$ could have an approximation ratio of $\frac{8k + 2}{5k + 2}$, which approaches $8/5 = 1.6$. 

15
Figure 4: Examples of the digraphs $H_k$ used to lower bound the integrality gap of MSCS and SSC. (a) One instance of the digraph $H$ (b) $H_6$ (c) $H_9$

5 Lowerbound on Integrality Gap of MSCS and SSC

We prove our lower bound on the integrality gap by giving a family of instances of the MSCS that approach this integrality gap. Let $H$ be the digraph defined on vertex set $\{s, t, u, v\}$ and arc set $\{su, ut, tv, vs, uv, vu\}$ (shown in Figure 4 (a)). We define $H_k$ for $k \geq 2$ to be a digraph constructed as follows. Construct $k$ copies of $H$ each with vertices $\{s_i, t_i, u_i, v_i\}$. Then combine each $t_i$ with $s_{i+1}$ for $1 \leq i < k$ and $t_k$ with $s_1$. Examples of $H_k$ are given in Figure 4.

Lemma 12. The optimal integer program solution of $H_k$ costs at least $4k - 1$.

Proof. Let $H_{OPT}$ be an optimal solution to the integer program on $H_k$. Suppose an instance of $H$ in $H_k$ only has three arcs in common with $H_{OPT}$. Then either arcs $\{su, uv, vs\}$ or $\{tv, vu, ut\}$ are in $H_{OPT}$. Either way the instance of $H$ is internally disconnected in $H_{OPT}$. There cannot be two instances of $H$ with only three arcs in $H_{OPT}$, because this implies the solution is disconnected. Thus the cost of an optimal solution is at least $3 + 4(k - 1) = 4k - 1$.

Lemma 13. The optimal linear program solution of $H_k$ costs at most $3k$.

Proof. Consider the linear program solution of assigning $1/2$ to all arcs. This solution costs $3k$. If we show this solution is feasible, our lemma follows. We can show our solution is feasible by proving that there are two arc-disjoint paths between any pair of vertices. This is sufficient because this will guarantee at least two arcs cross every cut, each assigned value $1/2$.

We use the following property, which can be verified using Menger’s Theorem: If two arc-disjoint paths from $u$ to $v$ exist and two arc-disjoint paths from $v$ to $w$ exist, then two arc-disjoint paths from $u$ to $w$ exist. Using this property and the symmetry between $u$ and $v$ in $H$, it is sufficient to show two arc-disjoint paths exist from $s_i$ to $u_j$ and from $u_j$ to $s_i$ in $H_k$ for any $1 \leq i, j \leq k$. 

16
Let $C_1$ denote the cycle $s_1u_1s_2u_2\ldots s_ku_ks_1$. Let $C_2$ denote the cycle $s_1v_kv_kv_{k-1}\ldots s_2v_1s_1$. Note that $C_1$ and $C_2$ share no arcs. Then two arc-disjoint paths exist from $s_i$ to $u_j$ as follows: follow $C_1$ from $s_i$ to $u_j$, and follow $C_2$ from $s_i$ to $v_j$ then cross the arc $v_ju_j$. Similarly, two arc-disjoint paths exist from $u_j$ to $s_i$ as follows: follow $C_1$ from $u_j$ to $s_i$, and cross the arc $u_jv_j$ then follow $C_2$ from $v_j$ to $s_i$.

From Lemmas 12 and 13, the problem instance $H_k$ has integrality gap at least $\frac{4k-1}{3k}$. As $k$ approaches infinity this ratio approaches $4/3$. Theorem 3 follows.

6 Conclusion

We presented a novel method of approximating graph connectivity problems using their dual problem as a lower bound. By applying this method to Dual Power Assignment, we get $3/2$-approximation, improving on the previous best approximation of $11/7$. Our analysis was made simpler by viewing DPA as a special case of a new problem we introduced, Star Strong Connectivity. SSC is a generalization of MSCS and Dual Power Assignment. Applying our method to the general case of SSC gives a 1.6-approximation. As a consequence of our algorithms, we proved new upper bounds on the integrality gaps of these problems (improving the upper bounds of 2 for MSCS and 1.85 for Dual Power Assignment). In addition to this, we proved our approximation ratios are tight and that the integrality gap of MSCS and SSC is at least $4/3$.

Our Dual Fitting approach can likely be applied to other unweighted connectivity problems with cut-based linear programs. Further application of this method will likely yield new approximation algorithms and improved upper bounds on their integrality gaps. Another interesting extension of this work would generalize SSC to have costs on each star. Even when costs are constrained to be in $\{0, 1\}$, Weighted SSC is at least as hard to approximate as Set Cover. This follows from a very simple reduction that was observed in [12]. As a consequence, any work on approximating the weighted variant will at best achieve a logarithmic ratio. We believe the methods used in [22] will generalize to give Weighted SSC such a logarithmic approximation.

Acknowledgments. This research was supported in part by a College of Science Undergraduate Summer Research Award at the Illinois Institute of Technology. We thank Gruia Calinescu for his many valuable comments and fruitful discussions, which notably improved the paper.

References

[1] Harold N. Gabow, Michel X. Goemans, Éva Tardos, and David P. Williamson. Approximating the smallest k-edge connected spanning subgraph by LP-rounding. Netw., 53(4):345–357, July 2009.
[2] Wen-Tsuen Chen and Nen-Fu Huang. The strongly connecting problem on multihop packet radio networks. *Communications, IEEE Transactions on*, 37(3):293–295, Mar 1989.

[3] Mohammadtaghi Hajiaghayi, Nicole Immorlica, and Vahab S. Mirrokni. Power optimization in fault-tolerant topology control algorithms for wireless multi-hop networks. In *Proceedings of the 9th Annual International Conference on Wireless Mobile Computing and Networking*. 2003, pages 300–312. ACM Press, 2003.

[4] Ram Ramanathan and R. Rosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 404–413 vol.2, 2000.

[5] R. Wattenhofer, Li Li, P. Bahl, and Yi-Min Wang. Distributed topology control for power efficient operation in multihop wireless ad hoc networks. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1388–1397 vol.3, 2001.

[6] Chen Wang, Myung-Ah Park, James Willson, Yongxi Cheng, Andras Farago, and Weili Wu. On approximate optimal dual power assignment for biconnectivity and edge-biconnectivity. *Theoretical Computer Science*, 396(13):180 – 190, 2008.

[7] NhatX. Lam, TracN. Nguyen, MinKyung An, and DungT. Huynh. Dual power assignment optimization and fault tolerance in wsns. *Journal of Combinatorial Optimization*, 30(1):120–138, 2015.

[8] Paz Carmi and Matthew J. Katz. Power assignment in radio networks with two power levels. *Algorithmica*, 47(2):183–201, 2007.

[9] G. Calinescu and K. Qiao. Asymmetric topology control: Exact solutions and fast approximations. In *INFOCOM, 2012 Proceedings IEEE*, pages 783–791, March 2012.

[10] Gruia Calinescu. Approximate min-power strong connectivity. *SIAM J. Discrete Math.*, 27(3):1527–1543, 2013.

[11] A. Karim Abu-Affash, Paz Carmi, and Anat Parush Tzur. Dual power assignment via second Hamiltonian cycle. *CoRR*, abs/1402.5783, 2014.

[12] Gruia Călinescu. 1.61-approximation for min-power strong connectivity with two power levels. *Journal of Combinatorial Optimization*, pages 1–21, 2014.

[13] S. Khuller, B. Raghavachari, and N. Young. Approximating the minimum equivalent digraph. *SIAM Journal on Computing*, 24(4):859–872, 1995.

[14] Samir Khuller, Balaji Raghavachari, and Neal Young. On strongly connected digraphs with bounded cycle length. *Discrete Applied Mathematics*, 69(3):281 – 289, 1996.
A Proof of Claim 1

We give a procedures that will turn any instance of DPA on digraph $H$ into a bidirected instance of SSC, $(G,C)$, and the reverse direction. Our transformation has linear runtime and will not substantially alter the size of the problem instance. Then our equivalence will follow.

We first consider transforming an instance of DPA into SSC. Let $H_0$ be the digraph induced by assigning no vertices of $H$ high power. Then $H_0$ will only have zero cost arcs. Since instances of DPA are bidirected, no arcs cross between the strongly connected components of $H_0$. We then construct an instance of SSC with a vertex for each strongly connected component of $H_0$. For each vertex $v$ in $H_0$, we add a star with source at the strong component of $v$ and arcs going to each other strong component that $v$ has an arc to in $H$. Note that $G$ is bidirected. Feasible solutions to these DPA and SSC instances can be converted between the two while preserving objective as follows: Given a feasible solution to DPA, for each vertex assigned high power add the corresponding star to our SSC solution. Similarly, given
a feasible solution to SSC, we assign each vertex to high power when the corresponding star is in our SSC solution. This will produce a feasible DPA instance. Note these conversions will have equal objective since there is a one-to-one mapping between high power vertices and stars.

We then show a transformation for the reverse direction from a bidirected instance of SSC with digraph $G$ and star set $C$. Our instance of DPA will have a vertex $v_F$ for every star $F \in C$. Add zero-cost arcs between any two vertices $v_F$ and $v_{F'}$ if $F$ and $F'$ have the same source endpoint. For every pair of arcs, $uv$ and $vu$, in our bidirected $G$ and for every $uw \in F$ and $vu \in F'$, we add a one-cost arcs between to $v_F$ and $v_{F'}$. The resulting digraph will be bidirected, as is required for DPA. As with our previous transformation, there is a one-to-one relationship between high power vertices and stars. This relationship immediately gives a conversion between our feasible solutions that will maintain objective.

## B Proof of Claim 3

Consider some instance of SSC given by a digraph $G = (V,E)$ and a set of stars $C$. Let $R_{OPT} \subseteq C$ be the optimal solution to SSC. Let $x^*$ be the optimal solution to SSC Primal LP when restricted to $x_F \in \mathbb{Z}$. We then need to show that $|R_{OPT}| = \sum_{F \in C} x^*_F$.

First we show that $|R_{OPT}| \geq \sum_{F \in C} x^*_F$. Consider the vector $x$ produced by assigning all $F \in R_{OPT}$ value 1 and the rest value 0. Then $|R_{OPT}| = \sum_{F \in C} x_F$. Our inequality will follow if we show $x$ is a feasible solution to SSC Primal LP, since $x^*$ is the minimum feasible solution. From our construction, all $x_F \geq 0$. Further consider any cut $\emptyset \subset S \subset V$. Since $R_{OPT}$ produces a strongly connected spanning subgraph, some $F \in R_{OPT}$ crosses $S$. Since this $x_F = 1$, we know $\sum_{F \in \delta(S)} x_F \geq 1$. Thus $x$ is feasible.

Now we prove that $|R_{OPT}| \leq \sum_{F \in C} x^*_F$. We know that all $x^*_F \in \{0,1\}$ (if a larger $x^*_F$ exists, our objective is reduced by reducing it to $x^*_F = 1$ without effecting feasibility). Consider the set of stars $R = \{F| x^*_F = 1\}$. Then $|R| = \sum_{F \in C} x^*_F$. Our inequality will follow if we show $R$ is a feasible solution to SSC, since $R_{OPT}$ is the minimum feasible solution. We prove this by contradiction. Let $G'$ be the digraph induced by $R$ (ie $G' = (V, \bigcup_{F \in R} F)$). Assume $G'$ is not strongly connected. Then there exists $s, t \in V$ such that there is no $s,t$-path in $G'$. Consider the set $V_s \subseteq V$ of all vertices $u$ with a $s,u$-path. Note $t \notin V_s$. Then $V_s$ is a cut with no arcs or stars crossing it. However, this contradicts the fact that $x^*$ is feasible. Thus we can conclude $R$ is feasible. Claim 3 follows.