The Primal-Dual Method

Eric Bauer and **Martin Zinkevich**

April 17, 2000

Background

We've seen a variety of approximation algorithms that relax a hard problem into a linear program and solve optimally for the relaxation.

Given the a solution to the LP, we've seen greedy and randomized methods for rounding the LP solution back into the original integer domain.

Problems we solved approximately using LPs:

- Vertex Cover
- Set Cover
- MAX-SAT
- Routing

Primal-Dual Method

Today we will explore a different method for approximating problems that can be relaxed from integer programs to linear programs.

Strangely enough, we won't bother finding an optimal solution for the relaxation.

Instead, we will find a second transformation of the problem: the dual of the linear program.

With this and the original LP relaxation, we find an approximate solution to the original problem using a clever iterative method.

Bounding an LP

Suppose our relaxed linear program is

$$\begin{array}{rll} \text{minimize} & z = \sum_{i=1}^{n} c_i x_i & \\ \text{when} & \sum_{i=1}^{n} a_{ij} x_i \geq b_j & \text{for } 1 \leq j \leq m \\ \text{and} & x_i \geq 0 & \text{for } 1 \leq i \leq n \end{array}$$

Let's attempt to find a good lower bound on z. One way to do this is to find some multiple of a constraint that has to be smaller than z.

Example:

$$\begin{array}{rll} \text{minimize} & z = x_1 + 12x_2 - 3x_3 & \\ \text{when} & 5x_1 + 2x_3 \geq 4 & \\ & 2x_1 + 16x_2 - 8x_3 \geq 2 & \\ \text{and} & x_i \geq 0 & \text{for } 1 \leq i \leq n \end{array}$$

If we halve the second constraint, we find

$$1 \leq x_1 + 8x_2 - 4x_3 \leq x_1 + 12x_2 - 3x_3 = z$$

More complex lower bounds

More generally, we can construct a linear combination of constraints to produce a lower bound.

$$
\begin{array}{rcl}
y_1 \left(\sum_{i=1}^{n} a_{i1} x_i \right) & \geq & y_1 b_1 \\
y_2 \left(\sum_{i=1}^{n} a_{i2} x_i \right) & \geq & y_2 b_2 \\
& \vdots & \\
y_m \left(\sum_{i=1}^{n} a_{im} x_i \right) & \geq & y_m b_m \\
\hline
\sum_{j=1}^{m} y_j \sum_{i=1}^{n} a_{ij} x_i & \geq & \sum_{j=1}^{m} y_j b_j
\end{array}
$$

Note that

$y_j \geq 0$ for $1 \leq j \leq m$

A new linear program!

How do we make this a lower bound for z? If we flip the sums, this becomes clear. We'd like

$$\sum_{j=1}^{m} y_j b_j \leq \sum_{i=1}^{n} \left(\sum_{j=1}^{m} y_j a_{ij} \right) x_i \leq \sum_{i=1}^{n} c_i x_i = z$$

Since the x_i's are positive, this has to hold when

$$\sum_{j=1}^{m} y_j a_{ij} \leq c_i \text{ for } 1 \leq i \leq n$$

Moreover, we'd like the lower bound to be as tight as possible. *In other words, we want to maximize its value.*

We've now constructed a new linear program, which is the dual of the original LP (called the primal).

The Duality Theorem

Primal:

$$\begin{array}{rll} \text{minimize} & z = \sum_{i=1}^{n} c_i x_i & \\ \text{when} & \sum_{i=1}^{n} a_{ij} x_i \geq b_j & \text{for } 1 \leq j \leq m \\ \text{and} & x_i \geq 0 & \text{for } 1 \leq i \leq n \end{array}$$

Dual:

$$\begin{array}{rll} \text{maximize} & z^* = \sum_{j=1}^{m} b_j y_j & \\ \text{when} & \sum_{j=1}^{m} a_{ij} y_j \leq c_i & \text{for } 1 \leq i \leq n \\ \text{and} & y_j \geq 0 & \text{for } 1 \leq j \leq m \end{array}$$

The Duality Theorem (Gale, Kuhn, and Tucker 1951):
the primal has an optimal solution iff the dual has an optimal solution, and the objective functions are equivalent in such a case, i.e.

$$z = \sum_{i=1}^{n} c_i x_i = \sum_{j=1}^{m} b_j y_j = z^*$$

when the x_i's and y_j's are optimal solutions.

Slackness

Let $\widehat{x}_1, \widehat{x}_2, \ldots \widehat{x}_n$ be an feasible solution for the primal, and let $\widehat{y}_1, \widehat{y}_2, \ldots \widehat{y}_m$ be an feasible solution for the dual. Consider a $\widehat{y}_s$ such that $\widehat{y}_s > 0$, and suppose there's still a little slack in the corresponding primal constraint: that is, $\sum_{i=1}^{n} a_{is}\widehat{x}_i > b_s$. Then

$$\begin{aligned}
\sum_{i=1}^{n} c_i \widehat{x}_i &\geq \sum_{i=1}^{n} \left(\sum_{j=1}^{m} \widehat{y}_j a_{ij} \right) \widehat{x}_i \\
&= \sum_{j=1}^{m} \widehat{y}_j \left(\sum_{i=1}^{n} a_{ij} \widehat{x}_i \right) \\
&= \left(\sum_{j \neq s} \widehat{y}_j \sum_{i=1}^{n} a_{ij} \widehat{x}_i \right) + \widehat{y}_s \sum_{i=1}^{n} a_{is} \widehat{x}_i \\
&> \sum_{j=1}^{m} b_j \widehat{y}_j
\end{aligned}$$

So $\sum_{i=1}^{n} c_i \widehat{x}_i \neq \sum_{j=1}^{m} b_j \widehat{y}_j$; therefore, by the Duality Theorem, these can't be optimal solutions.

Complementary Slackness

We can make a similar argument to show that optimal solutions can't have any slack in a dual constraint when the corresponding value in the primal is positive. So if $x_1^*, x_2^*, \ldots x_n^*$ and $y_1^*, y_2^*, \ldots y_m^*$ are optimal solutions for the primal and dual, then

$$x_i^* > 0 \quad \text{implies} \quad \sum_{j=1}^{m} a_{ij} y_j^* = c_i$$

$$y_j^* > 0 \quad \text{implies} \quad \sum_{i=1}^{n} a_{ij} x_i^* = b_j$$

These conditions are respectively called the primal and dual complementary slackness conditions.

Complementary Slackness cont.

Suppose feasible solutions $\hat{x}_1, \hat{x}_2, \ldots \hat{x}_n$; $\hat{y}_1, \hat{y}_2, \ldots \hat{y}_m$ satisfy

$$\hat{x}_i > 0 \quad \text{implies} \quad \sum_{j=1}^{m} a_{ij}\hat{y}_j = c_i$$

$$\hat{y}_j > 0 \quad \text{implies} \quad \sum_{i=1}^{n} a_{ij}\hat{x}_i = b_j$$

$$\begin{aligned}
\sum_{i=1}^{n} c_i\hat{x}_i &= \sum_{i:\hat{x}_i>0} c_i\hat{x}_i = \sum_{i:\hat{x}_i>0} \left(\sum_{j=1}^{m} \hat{y}_j a_{ij}\right)\hat{x}_i \\
&= \sum_{i:\hat{x}_i>0} \left(\sum_{j:\hat{y}_j>0} \hat{y}_j a_{ij}\right)\hat{x}_i \\
&= \sum_{j:\hat{y}_j>0} \hat{y}_j \left(\sum_{i:\hat{x}_i>0} a_{ij}\hat{x}_i\right) = \sum_{j:\hat{y}_j>0} \hat{y}_j b_j \\
&= \sum_{j=1}^{m} b_j\hat{y}_j
\end{aligned}$$

$\Rightarrow$ The solutions are optimal!

Great. So what?

If we found a solution that satisfied all the complementary slackness conditions, we'd have an optimal solution for the LPs.

But we don't need an optimal solution for the LPs. We need a good solution for the original integer program, which is probably not optimal for either the primal or dual LP.

This suggests two ideas:

1. Ignore some complementary slackness conditions.
   Since the primal conditions are simpler over the variables in the original problem, let's just throw out the dual complementary slackness conditions.

2. Search directly for an integer solution. We can use the original primal/dual constraints as well as the primal slackness conditions to guide the search!

The Primal-Dual Method

1. Find initial feasible dual LP solution $\mathbf{y}$, and set initial integer primal variables $\mathbf{x}$ to 0.

2. Test dual constraints to see which satisfy

   $$\sum_{j=1}^{m} a_{ij} y_j = c_i$$

   Satisfying dual constraints correspond exactly to x_i's that we can increase above zero (by primal slackness).

3. Try out all possible integer solutions that only increase x_i's that passed the above test. If we find a feasible solution to the original integer program, terminate.

4. Otherwise, pick some violated primal constraint $\sum_{i=1}^{n} a_{ik} x_i < b_k$ for one of the solutions we tried. Raise y_k until $\sum_{j=1}^{m} a_{ij} y_j = c_i$ for some i.

5. Goto 2.

Discussion

The primal-dual method gradually considers larger sets of integer program solutions until a feasible integer solution is found or the dual can't be increased anymore.

The search is guided by the LP constraints without being forced to solve the LP.

The result will be an approximately good solution for the original integer program.

But how good?

Performance Guarantees

Suppose we find a feasible integer program solution. Call it $\widehat{x}_1, \widehat{x}_2, \ldots \widehat{x}_n$, and call the dual solution we ended up with after running the algorithm $\widehat{y}_1, \widehat{y}_2, \ldots \widehat{y}_m$.

$$
\begin{aligned}
z &= \sum_{i=1}^{n} c_i \widehat{x}_i = \sum_{i:\widehat{x}_i > 0} c_i \widehat{x}_i \\
&= \sum_{i:\widehat{x}_i > 0} \left(\sum_{j=1}^{m} a_{ij} \widehat{y}_j \right) \widehat{x}_i \\
&= \sum_{j=1}^{m} \left(\sum_{i:\widehat{x}_i > 0} a_{ij} \widehat{x}_i \right) \widehat{y}_j \geq \sum_{j=1}^{m} b_j \widehat{y}_j
\end{aligned}
$$

The last inequality is due to the primal constraints. Suppose that we could quantify how much bigger the lhs was than the rhs.

Performance Guarantees cont.

More formally, suppose we knew that whenever we have $\widehat{y}_j > 0$, then $\sum_{i=1}^{n} a_{ij}\widehat{x}_i \leq \alpha b_j$ for some $\alpha \geq 1$. Then

$$
\begin{aligned}
z &= \sum_{i=1}^{n} c_i\widehat{x}_i = \sum_{i:\widehat{x}_i>0} c_i\widehat{x}_i \\
&= \sum_{i:\widehat{x}_i>0} \left(\sum_{j=1}^{m} a_{ij}\widehat{y}_j\right)\widehat{x}_i \\
&= \sum_{j=1}^{m} \widehat{y}_j \left(\sum_{i:\widehat{x}_i>0} a_{ij}\widehat{x}_i\right) = \sum_{j:\widehat{y}_j>0} \widehat{y}_j \left(\sum_{i:\widehat{x}_i>0} a_{ij}\widehat{x}_i\right) \\
&\leq \alpha \sum_{j=1}^{m} \widehat{y}_j b_j \\
&\leq \alpha OPT^{Dual} = \alpha OPT^{Primal} \\
&\leq \alpha OPT^{IP}
\end{aligned}
$$

$\Rightarrow$ We've found an α-approximation!

Unresolved questions

- When can we find an α bound for coefficients of the non-zero dual variables?
- How tight can the approximations be?
- How do we pick a direction of increase for the dual?
- What problems can we solve using this method?

Looking at some specific examples will help answer these questions.

Weighted Set Cover

Weighted Set Cover

Inputs:	Set E
	Sets $S_i \subset E$
	Associated weights to sets w_i

Output: Minimum set of sets that "cover" all the elements in E, i.e. that the union of the sets is E.

Hitting Set

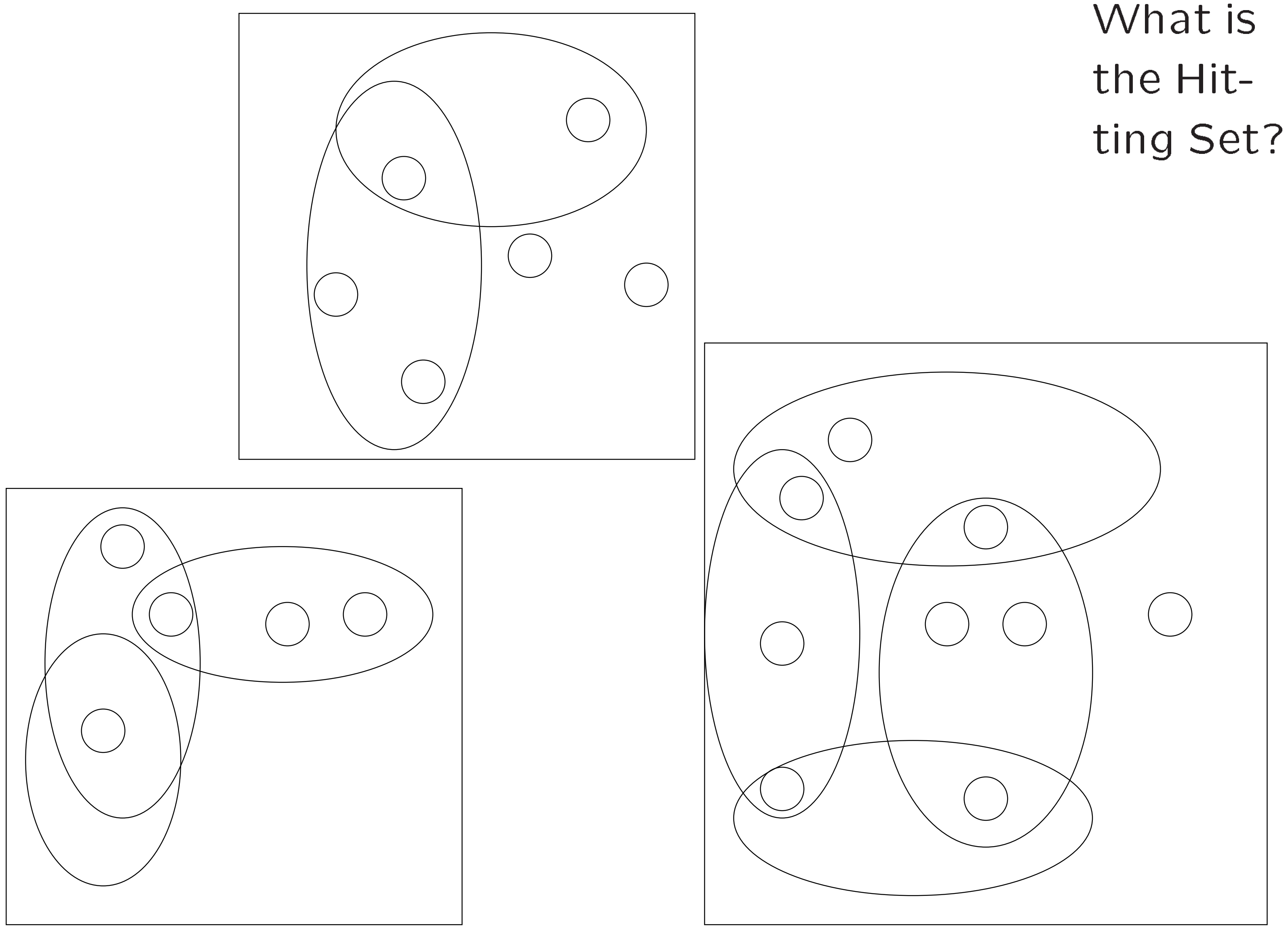
Inputs:

- Ground Set E
- Costs c_e, $\forall e \in E$
- Sets $T_i \subset E$, $i = 1, \ldots, p$

Output: Minimum cost $A \subset E$ s.t. $A \cap T_i \neq \emptyset$
i.e. - A set of elements which "cover" every set.
Called the **Hitting Set**.

Isomorphic problems: sets become points and points become sets.

Example



What is the Hitting Set?

Primal-Dual Method

One way of viewing the Primal-Dual Method is a solution to the hitting set problem. Other problems are reduced to the hitting set problem.

Example: Vertex Cover

Given Graph $G = (V, E)$

Given weights w_i on $i \in V$, find a min-cost subset S of V such that at least one endpint of every edge is in S.

Ground set: V
Costs: $w_i, i \in V$
Sets: $\{u, v\}$, where $(u, v) \in e$

Shortest $s-t$ Path

Ground Set: E
Costs: $c_e \geq 0$, $\forall e \in E$
Sets: $\delta(S)$, $\forall S \in V$ such that $s \in S, t \notin S$
$\delta(S) \equiv \{(u,v) \in E : u \in S, v \notin S\}$ (The cut between S and the rest.

In English: a path must cross every cut between its endpoints.

Hitting Set: Linear Integer Program

Variables: $x_e = 0$ if element e is not in the hitting set.
$x_e = 1$ if element e is in the hitting set.

Cost: $\sum c_i x_i$ Sum of the weights

Restrictions: $\sum_{e \in T_i} x_e \geq 1$ for $i = 1, \ldots, p$
$x_e \in 0, 1$

The dual of the relaxation:

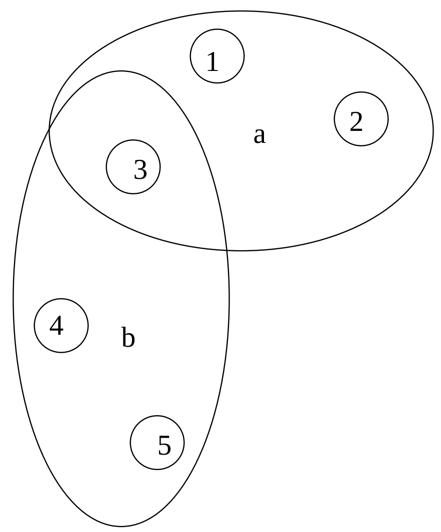
Maximize $\sum y_i$
subject to: $\sum_{i: e \in T_i} y_i \leq c_e$ for all $e \in E$
$y_i \geq 0$

Primal slackness:

$x_e > 0 \Rightarrow \sum_{i: e \in T_i} y_i = c_e$

What is y_i?

$y_a \leq c_1$
$y_a \leq c_2$
$y_a + y_b \leq c_3$
$y_b \leq c_4$
$y_b \leq c_5$



y_1 is how expensive T_1 is to hit. y_2 is how expensive T_2 is to hit. So y_1+y_2 is a minimum on the cost of the hitting set. If $y_i = 0$, that means we get T_i for free!

Algorithm(Bar-Yehuda, Even)

$y \leftarrow 0$
$A \leftarrow \emptyset$ (so all $x_e = 0$)
While A is not feasible:

Pick a violated set T_j
Increase y_j until $\sum_{i:e \in T_i} y_i = c_e$ for some $e \in T_j$
$A \leftarrow A \cup \{e\}$

Note: $x_e > 0 \Rightarrow \sum_{i:e \in T_i} y_i = c_e$ at all times.

Vertex Cover Translation

$y \leftarrow 0$

$A \leftarrow \emptyset$ Begin with empty set

While A is not feasible: If not a vertex cover

> Pick a violated set T_j Choose an uncovered edge
>
> Increase y_j until $\sum_{i:e \in T_i} y_i = c_e$ for some $e \in T_j$ Choose an endpoint
>
> $A \leftarrow A \cup \{e\}$ Add it to vertex cover

Familiar?

Analysis of the Algorithm

Let A_f denote the output of the algorithm.

$$\begin{aligned}\sum_{e\in A_f} c_e &= \sum_{e\in A_f}\sum_{i:e\in T_i} y_i \\ &= \sum_i |A_f\cap T_i| y_i\end{aligned}$$

If we can ensure that $y_i > 0$ implies $|A_f \cap T_i| \le \alpha$, then

$\sum_{e\in A_f} c_e \le \alpha \sum_i y_i \le \alpha OPT$

i.e., we have an α-approximation algorithm.

Example For vertex cover, $|T_i| = 2$, so $|A_f \cap T_i| \le 2$, which implies a 2-approximation alg. (Bar-Yehuda, Even).

Algorithm, Version Two

$y \leftarrow 0$
$A \leftarrow \emptyset$
$l \leftarrow 0$
While A is not feasible:

$l \leftarrow l + 1$
Pick a violated set T_j
Increase y_j until $\sum_{i:e \in T_i} y_i = c_e$ for some $e_l \in T_j$
$A \leftarrow A \cup \{e\}$

For $j \leftarrow l$ downto 1

If $A - e_j$ feasible $A \leftarrow A - e_j$

NOTE: Last part does not invalidate primal slackness

Let's see how this works on shortest path.

Minimal Hitting Set

A minimal hitting set A is a set for which no $A' \subset A$ is a hitting set.

What this final step gives us is a minimal hitting set.

A minimal hitting set in terms of shortest path is a simple path.

Proof of Minimal Solution

Proof:
Suppose A is the result of the original algorithm. Suppose there exists some proper subset $S \subset A$. If an element a in $A - S$ is reached before any element is removed, that element will be removed(because the resulting set will be a superset of S, therefore a hitting set). If an element remains in the set, it is in every hitting set which is a subset of A. After this first element a is removed, hereafter the set $A' \equiv A - \{a\}$ is our basis. If an element is passed over, it is in every hitting set which is a subset of A'. Also, all elements previously passed over are in every hitting set which is a subset of A', because they are also subsets of A.

Minimal Augmenting Set

Consider a set A. B is a minimal augmenting set of A iff $A \cup B$ is a hitting set, and if there exists no proper subset $B' \subset B$ where $A \cup B'$ is a hitting set.

Because we remove elements in the reverse order, there will exist some minimal augmenting set B such that $A \cup B$ will be a superset of A_f.

Analysis of Algorithm II

Consider the algorithm at some state, where A is the current set. Define T(A) to be the invalidated set to be chosen by the algorithm. If for **all** minimal augmenting sets B:

$$|B \cap T(A)| \leq \beta$$

then we have a β-approximation algorithm.
Proof:
If y_j is nonzero, then at some point it was a violated set and was selected by the algorithm. Define A to be the hitting set in progress at that point, and B a minimal augmenting set such that $B \cup A \supseteq A_f$. Then:
$A_f \cap T_j \subseteq (A \cup B) \cap T_j$
$A_f \cap T_j \subseteq (A \cap T_j) \cup (B \cap T_j)$
$A_f \cap T_j \subseteq \emptyset \cup (B \cap T_j)$
$A_f \cap T_j \subseteq B \cap T_j$
$|A_f \cap T_j| \leq |B \cap T_j| \leq \beta$

Since slackness conditions still satisfied, we have a β-approximation.

Shortest s-t Path

Since we defined $T(A)$ to be the cut where all reached elements are on one side and all unreached elements are on the other side, a minimal augmenting set can cross it only once. If it crossed it twice, it would form a cycle. Therefore, $|B \cap T(A)| \leq 1$, and this becomes a 1-approximation, or in English, a solution to the problem.

Other Uses

$\alpha = 1$	
Shortest s-t path	Dijkstra '59
Min-cost branching	Edmonds '67
$\alpha = 2$	
Vertex cover	Bar-Yehuda, Even '81
Steiner tree	various
Generalized Steiner tree	Agrawal, Klein, Ravi '91
Matching(w/ triangle inequality)	Goemans, W '92
T-joins	
Non-fixed point-to-point connection	
Partition into trees/paths/cycles of at least k vertices	
Location/design problems	
Prize-collecting TSP	
Multicuts in trees	Garg, Vazirani, Yannakakis '93
Other α	
Survivable network design	W, Goemans, Mihail, Vazirani '93
	Goemans, Goldberg, Plotkin, Shmoys, Tardos, W '94

Conclusion

- The Primal-Dual Method is derived from Linear Programming
- It is very useful in approximation algorithms
- It gives an approximate solution to hitting set
- It gives clear bounds given properties of the problem