A Note on the Connection Between the Primal-Dual and the A* Algorithm

Xugang Ye, Johns Hopkins University, USA
Shih-Ping Han, Johns Hopkins University, USA
Anhua Lin, Middle Tennessee State University, USA

ABSTRACT

The primal-dual algorithm for linear programming is very effective for solving network flow problems. For the method to work, an initial feasible solution to the dual is required. In this article, we show that, for the shortest path problem in a positively weighted graph equipped with a consistent heuristic function, the primal-dual algorithm will become the well-known A* algorithm if a special initial feasible solution to the dual is chosen. We also show how the improvements of the dual objective are related to the A* iterations.

Keywords: A* Algorithm, Primal-Dual, Shortest Path

INTRODUCTION

The primal-dual algorithm (Bertsimas & Tsitsiklis, 1997; Dantzig, Ford, & Fulkerson, 1956; Papadimitriou & Steiglitz, 1998) for linear programming is very effective for solving network flow problems. Despite the name, the method that we discuss in this article is not to be confused with the currently well-known primal-dual interior methods. The latter were originated by Karmarkar’s seminal paper (Karmarkar, 1984) and they enjoy polynomial time-complexity for solving general linear programming problems (Wright, 1997) and have been generalized for some convex conic optimization problems like second-order cone programming and semi-definite programming (Boyd & Vandenberghe, 2004). The primal-dual algorithm discussed in this article starts from an initial dual feasible solution and iteratively improves the solution until a primal feasible solution, determined by the current dual solution, is found such that the pair satisfies the complementary slackness conditions (Bertsimas & Tsitsiklis, 1997). In this article, we consider the primal-dual algorithm for the shortest path problem in a positively weighted graph equipped with extra information and study the consequence of choosing a proper initial feasible solution to the dual.

DOI: 10.4018/joris.2010101305
The key point of this article is to utilize the extra information, or heuristic in the language of the field of artificial intelligence, of the graph to construct an initial feasible solution to the dual. We show that the well-known A* algorithm (Hart, Nilsson, & Rapheal, 1968; Hart, Nilsson, & Rapheal 1972; Nilsson, 1980; Pearl, 1984) can be derived. This derivation actually goes directly from the primal-dual algorithm to the A* algorithm. Furthermore, we show how the improvements of the dual objective are related to the A* iterations.

This article is organized as follows. We first set up the problem domain and introduce the A* algorithm and the primal-dual algorithm. We then use the heuristic to construct an initial feasible solution to the dual and propose a best-first search (Pearl, 1984) version of the primal-dual algorithm. We show that this version of the primal-dual algorithm behaves essentially the same as the A* algorithm that uses the same heuristic. Finally, we provide additional discussions.

BACKGROUND

We consider a directed, positively weighted simple graph denoted as \( G = (V, A, W, \delta, b) \), where \( V \) is the set of nodes, \( A \) is the set of arcs, \( W: A \to R \) is the weight function, \( \delta > 0 \) is a constant such that \( \delta \leq W(a) < +\infty \) for all \( a \in A \), and finally \( b \) is a constant integer such that \( 0 < b < |V| \) and \( |\{v \mid (u, v) \in A \text{ or } (v, u) \in A\}| \leq b \) for all \( u \in V \). Suppose we want to find a shortest \( s-t \) (directed) path in \( G \), where \( s \in V \) is a specified starting node and \( t \in V \) is a specified terminal node. Further suppose a heuristic function exists \( h: V \to R \) such that \( h(v) \geq 0 \) for all \( v \in V \), \( h(s) = 0 \), and \( W(u, v) + h(v) \geq h(u) \) for all \( (u, v) \in A \). \( h \) is called consistent heuristic. According to Hart et al. (1968), Hart et al. (1972), Nilsson (1980), and Pearl (1984), the A* algorithm that uses such \( h \) is complete, that is, it can find a shortest \( s-t \) path in \( G \) as long as there exists an \( s-t \) path in \( G \). The algorithm can be stated as follows. It searches from \( s \) to \( t \).

The A* Algorithm

Notations:
- \( h \): heuristic
- \( O \): Open list
- \( E \): Closed list
- \( d \): distance label
- \( f \): node selection key
- \( pred \): predecessor

Steps:
Given \( G, s, t, \) and \( h \)
Step 1. Set \( O = \{s\}, d(s) = 0, \) and \( E = \phi \).
Step 2. If \( O = \phi \) and \( t \notin E \), then stop (there is no \( s-t \) path); otherwise, continue.
Step 3. Find \( u = \arg \min_{v \in O} f(v) = d(v) + h(v) \).
   Set \( O = O \setminus \{u\} \) and \( E = E \cup \{u\} \).
   If \( t \in E \), then stop (a shortest \( s-t \) path is found); otherwise, continue.
Step 4. For each node \( v \in V \) such that \( (u, v) \in A \) and \( v \notin E \),
   if \( v \notin O \), then
      set \( O = O \cup \{v\} \), \( d(v) = d(u) + W(u, v) \), and \( pred(v) = u \);
   otherwise,
      if \( d(v) > d(u) + W(u, v) \), then
         set \( d(v) = d(u) + W(u, v) \) and \( pred(v) = u \).
Go to Step 2.

In particular, when \( h = 0 \), the A* algorithm stated above reduces to the Dijkstra’s algorithm (Ahuja, Magnanti, & Orlin, 1993; Dijkstra, 1959; Papadimitriou & Steiglitz, 1998). For convenience, for any two nodes \( u \in V \) and \( v \in V \), let \( dist(u, v) \) denote the distance from \( u \) to \( v \) in \( G \). That is, if there is no \( u-v \) path in \( G \), we define \( dist(u, v) = +\infty \); otherwise, we define \( dist(u, v) \) to be the length of a shortest \( u-v \) path in \( G \). According to Hart et al. (1968) and Pearl (1984), a central property, called strong optimality, of the A* algorithm stated above is \( d(u) = dist(s, u) \) when \( u \in E \). If \( G \) is a large and sparse finite graph in the sense that \( b << |V| \), then the algorithm can be efficiently implemented by storing \( G \) in the form of adjacency lists (Cormen, Leiserson, Rivest, & Stein, 2001) and maintaining the Open list \( O \) as a binary heap,
Related Content

**Equal Pricing Strategies in a Dual Channel Supply Chain**
[www.igi-global.com/article/equal-pricing-strategies-dual-channel/58894?camid=4v1a](www.igi-global.com/article/equal-pricing-strategies-dual-channel/58894?camid=4v1a)

**Retailer’s Pricing and Lot Sizing Policy for Non Deteriorating Items with Constant Demand Rate Under the Condition of Permissible Delay in Payments**
[www.igi-global.com/article/retailer-pricing-lot-sizing-policy/62260?camid=4v1a](www.igi-global.com/article/retailer-pricing-lot-sizing-policy/62260?camid=4v1a)