

Formal Primal-Dual Algorithm Analysis

Mohammad Abdulaziz   

King's College London, UK

Thomas Ammer   

King's College London, UK

Christoph Madlener 

Technische Universität München, Germany

Abstract

We present an ongoing effort to build a framework and a library in Isabelle/HOL for formalising primal-dual arguments for the analysis of algorithms. We discuss a number of example formalisations from the theory of matching algorithms, covering classical algorithms like the Hungarian Method, widely considered the first primal-dual algorithm, and modern algorithms like the AdWords algorithm, which models the assignment of search queries to advertisers in the context of search engines.

2012 ACM Subject Classification Theory of computation → Discrete optimization; Theory of computation → Invariants; Theory of computation → Program verification

Keywords and phrases Bipartite Matching, Graph Algorithms, Isabelle/HOL, Formal Verification

Digital Object Identifier 10.4230/LIPIcs.ITP.2026.33

Category Short Paper

Supplementary Material

Software (Formal Mathematical Library): <https://zenodo.org/records/20370764>

Funding *Mohammad Abdulaziz:* Supported by the Renaissance Philanthropy Grant Co-Pilots for Isabelle and the Royal Society International Exchange grant Formal Verification of Practical Nominal Unification Algorithms (IES-R3-243187).

Thomas Ammer: Supported by an ECR Travel Grant from the London Mathematical Society (LMS) to attend ITP.

Declaration about GenAI/LLM use No AI was involved in this formalisation project.

1 Introduction

The primal-dual paradigm for analysing algorithms is one of the most successful. Its history spans more than 70 years, with the Hungarian Method [27] being one of the first algorithms for solving weighted bipartite matchings to follow this paradigm. Since then, the paradigm has been used to design algorithms for problems in combinatorial optimisation, including Edmonds' algorithm for weighted matching in general graphs [16], all the way to modern analyses of online matching algorithms [13] and some of the fastest algorithms for solving flow problems [12]. In addition to exact algorithms, the primal-dual approach is also cornerstone for the design for a majority of approximation algorithms for optimisation problems, like MaxSAT, set cover, Steiner trees, etc.(cf. Part II of Vazirani's seminal book on approximation algorithms, which is dedicated to the primal-dual method).

In this work, we present a series of formal analyses of primal-dual matching algorithms. Our analyses cover a range of primal-dual algorithms: from the Hungarian Method (HM), arguably the first primal-dual algorithm, to probabilistic arguments used to analyse online algorithms, including the well-known Adwords algorithm [31] and the RANKING algorithm [25]. Our longer term goal is to create a formal library of lemmas and reasoning principles that aid in the analysis of primal-dual algorithms.



© Mohammad Abdulaziz, Thomas Ammer, and Christoph Madlener;
licensed under Creative Commons License CC-BY 4.0

17th International Conference on Interactive Theorem Proving (ITP 2026).

Editors: Ekaterina Komendantskaya and Tobias Nipkow; Article No. 33; pp. 33:1–33:9

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Theory Background. We assume the reader to be familiar with basic graph theory, such as vertices $\mathcal{V}$, edges $\mathcal{E}$ and paths in graphs, bipartiteness, and adjacency and incidence matrices encoding graphs. Because it eases the formalisation, we identify a graph with its edges and define $\mathcal{V} = \bigcup \mathcal{E}$. A *matching* $\mathcal{M}$ in a graph $\mathcal{E}$ is a vertex-disjoint subset of the edges in $\mathcal{E}$. Matchings covering all vertices are *perfect*. We search for matchings that are optimum w.r.t. an optimisation objective e.g. the maximisation/minimisation of accumulated real weights.

In a primal-dual (PD) method, we maintain an upper bound for the value of the solution, and finally obtain a solution whose value equals the final upper bound (or is close to it), which implies optimality (or sufficient proximity). We encode the optimisation problem into linear (in-)equalities alongside a linear optimisation objective (*linear program/LP*), e.g. $\max\{c^T x \mid Ax \leq b, x \geq 0\}$ in linear algebra notation. The matrix A is usually the incidence matrix, a primal solution is a binary vector $x_{\mathcal{M}}$ encoding a matching, and a dual solution π is a $|\mathcal{V}|$ -dimensional vector, which encodes a function $\pi : \mathcal{V} \rightarrow \mathbb{R}$ (“potential”).

For $\max\{c^T x \mid Ax \leq b, x \geq 0\}$, the *primal*, there is a *dual* $\min\{b^T y \mid A^T y \geq c, y \geq 0\}$. Vectors x and y satisfying these constraints are *feasible primal* and *dual solutions*, for which $c^T x \leq b^T y$ holds (*weak duality WD*). Furthermore, if $((1 - \delta)A\hat{x} - b)^T \hat{y} = 0$ and $(1 - \delta)(A^T \hat{y} - c)^T \hat{x} = 0$, then $(1 - \delta)c^T \hat{x} = b^T \hat{y}$ and hence both $(1 - \delta)\hat{x}$ and $\hat{y}$ are optimum solutions to the respective LPs (*complementary slackness CS*). If $\delta = 0$, $\hat{x}$ is an exact optimum, otherwise $\hat{x}$ satisfies an approximation guarantee. For further details, e.g. other LP types and their versions of WD and CS, we refer to the literature [34, 26].

A PD method starts with a feasible dual solution and a primal candidate solution which together satisfy CS [20] (scaled by $1 - \delta$ for approximation). It iteratively changes both solutions to bring the primal closer to feasibility while maintaining CS. These steps are *primal-dual adjustments (PDAs)*. When the primal solution becomes feasible, it is also optimum (or approximately optimum if $\delta \neq 0$).

Formalisation. Our work here focuses on presenting aspects related to formal reasoning about primal-dual analyses. The main design choice regarding that is the representation of LPs: we reuse matrices [37] others used for verifying LP theory, e.g. strong duality [33] and the simplex algorithm [36]. The mathematical core of the arguments, namely, CS and WD, is purely matrix-based and then connected to graphs by translating lemmas, which connect matchings and potentials on graphs to feasible LP solutions over matrices and vectors. The rest of the paper focuses on the reasoning of primal-dual analyses.

Other design choices peripheral to primal-dual reasoning include our definition of matching. We reuse matching-related Isabelle/HOL formalisations [7, 6], which already contain many of the graph-theoretic concepts introduced above. Another decision is the approach used to model and verify algorithms, where we reuse an approach by other authors [1, 2, 3, 5, 7]. In summary, we formalise deterministic algorithms as functional programs, i.e. loops as recursive functions and program states (collections of the variables a program uses) as records. We prove *invariants* (properties of the state) for the initial state, and that they are preserved by the iterations, which shows them for the final state. For probabilistic algorithms, we use the existing Isabelle/HOL formalisation of the Giry monad [14] to model and reason about them. Algorithms have subprocedures that we assume and specify with Isabelle/HOL’s *locales* [9]. These are contexts that fix mathematical entities (‘locale constants’) and assume properties. Within the context, these are available and they can be instantiated later. This is a *stepwise refinement* [40] where an instruction is decomposed into more detailed instructions.

■ **Algorithm 1** NAIVEMAXWEIGHTMATCHING ($\mathcal{E}$, $\mathcal{V} = L \cup R$, $w : \mathcal{E} \rightarrow \mathbb{R}_0^+$).

```

1 Initialise  $\pi(v) = \max\{w(e) \mid e \in \Delta(\{v\})\}$  for  $v \in L$  and  $\pi(v) = 0$  for  $v \in R$ ;
2 while True do
3   if  $\exists$  matching  $\mathcal{M} \subseteq \mathcal{E}_\pi \cdot \{v.v \in \mathcal{V} \wedge \pi(v) > 0\} \subseteq \bigcup \mathcal{M}$  then return such an  $\mathcal{M}$ ;
4   else
5     find  $X$  where  $X \subseteq L$  or  $X \subseteq R$  with  $|X| > |\Gamma_\pi(X)|$  and compute
6      $\epsilon = \min(\{w_\pi(\{u, v\}) \mid u \in X \wedge v \notin \Gamma_\pi(X) \wedge w_\pi(\{u, v\}) > 0\} \cup \{\pi(v).v \in X\})$ ;
7     for  $x \in X$  do  $\pi(x) \leftarrow \pi(x) - \epsilon$ ; for  $x \in \Gamma_\pi(X)$  do  $\pi(x) \leftarrow \pi(x) + \epsilon$ ;

```

2 Naive Maximum Weight Bipartite Matching

We search for matchings $\mathcal{M} \subseteq \mathcal{E}$ maximising $w(\mathcal{M})$ and we derive a CS-based optimality criterion. (Maximum) integral solutions to $\max\{w^T x_{\mathcal{M}} \mid Ax_{\mathcal{M}} \leq 1, x_{\mathcal{M}} \geq 0\}$ encode a (max-weight) matching for $w : \mathcal{E} \rightarrow \mathbb{R}_0^+$ if A is the incidence matrix. The constraints $Ax_{\mathcal{M}} \leq 1$ and $x_{\mathcal{M}} \geq 0$ assert that $\mathcal{M}$ is a matching. The dual is $\min\{1^T \pi \mid A^T \pi \geq w, \pi \geq 0\}$, i.e. find π with minimum $\pi(\mathcal{V})$ s.t. $\forall \{u, v\} \in \mathcal{E}. \pi(u) + \pi(v) \geq w(\{u, v\})$ and $\forall v \in \mathcal{V}. \pi(v) \geq 0$. Such a π is a *feasible potential*, and w_π , defined as $w_\pi(\{u, v\}) = \pi(u) + \pi(v) - w(e)$ for $\{u, v\} \in \mathcal{E}$, is the *slack*. For a potential π , v with $\pi(v) \neq 0$ is a *non-zero vertex*, and e with $w_\pi(e) = 0$ is *tight*. Tight edges form the *tight subgraph* $\mathcal{E}_\pi$. By $\Gamma_\pi(X)$ we denote those vertices $v \in \mathcal{V} \setminus X$ where there is $\{u, v\} \in \mathcal{E}_\pi$ with $u \in X$. Similarly, $\Delta_\pi(X)$ are the tight edges connecting X and $\Gamma_\pi(X)$. We use this machinery to prove a sufficient condition for the weight-maximality of a matching:

► **Lemma 1** (Weight-Maximality). *Assume (1) $\mathcal{M}$ is a matching in a graph $\mathcal{E}$, (2) π is a feasible vertex potential, (3) all edges in $\mathcal{M}$ are tight, i.e. $\mathcal{M} \subseteq \mathcal{E}_\pi$, and (4) all vertices for which $\pi(v) \neq 0$ are matched. Then, $\mathcal{M}$ is a max-weight matching for $\mathcal{E}$.*

The Algorithm. Algorithm 1 works on $\mathcal{E}$ bipartite over L and R and $w : \mathcal{E} \rightarrow \mathbb{R}_0^+$. After initialising π to a feasible potential, it tries to find a tight matching covering all non-zero vertices. If there is such a matching, this is returned as an optimum solution. Otherwise, it performs a PDA behind which the intuition is to move the primal (matching that matches as many non-zero vertices as possible) and the dual solution π closer together. If $\mathcal{E}$ is bipartite and there is no matching in $\mathcal{E}_\pi$ covering the non-zero vertices, we can find non-zero vertices X with $|X| > |\Gamma_\pi(X)|$, as used for the PDA. The main invariant of this algorithm is the feasibility of π (Invariant N1). We have another invariant N2 saying that the vertex potentials $\pi(v)$ are integer multiples of the same real constant α , ensuring termination for a specific class of edge weights. We need two lemmas for correctness:

► **Lemma 2.** *Let π be feasible for $\mathcal{E}$ and $w : \mathcal{E} \rightarrow \mathbb{R}_0^+$, and S be a set of vertices. Also, assume there is no $e \in \mathcal{E}$ with $e \subseteq S$, and $\epsilon \geq 0$, $\epsilon \leq w_\pi(e)$ for all e connecting S and $\mathcal{V} \setminus S \setminus \Gamma_\pi(S)$, and $\forall v \in S. \epsilon \leq \pi(v)$. Let $\pi'(v) = \pi(v) - \epsilon$ for $v \in S$, $\pi'(v) = \pi(v) + \epsilon$ for $v \in \Gamma_\pi(S)$, and $\pi'(v) = \pi(v)$ otherwise. The potential π' is feasible, and $\pi'(\mathcal{V}) = \pi(\mathcal{V}) + (|\Gamma_\pi(S)| - |S|) \cdot \epsilon$.*

Proof Ideas. Take $\{u, v\} \in \mathcal{E}$, we know $\pi(u) + \pi(v) \geq w(\{u, v\})$, and $\pi'(u) + \pi'(v) \geq w(\{u, v\})$ follows by a case analysis. Also $\pi'(v) \geq 0$ because $\pi'(v) \leq \pi(v)$ only if $v \in S$, and then, $\pi'(v) = \pi(v) - \epsilon \geq 0$. The statement on $\pi'(\mathcal{V})$ follows from the definition of π' . ◀

► **Lemma 3.** *If Algorithm 1 terminates on $\mathcal{E}$ bipartite over L and R with $w : \mathcal{E} \rightarrow \mathbb{R}_0^+$, it returns a max-weight matching. If all $w(e)$ are integer multiples of $\alpha \in \mathbb{R}$, it terminates.*

■ **Algorithm 2** HUNGARIANMETHOD ($\mathcal{E}, \mathcal{V} = L \cup R, w : \mathcal{E} \rightarrow \mathbb{R}$, initial mp-feasible π).

```

1 if  $|L| \neq |R|$  return infeasible; else initialise  $\mathcal{M} = \emptyset$ ;
2 while True do  $(flag, \pi', p) \leftarrow \text{PATHSEARCH}(\mathcal{E}, L, R, w, \mathcal{M}, \pi)$ ;
3   if  $flag = \text{dual-unbounded}$  then return infeasible
4   else if  $flag = L\text{-matched}$  then return  $\mathcal{M}$  else  $[\mathcal{M} \leftarrow \mathcal{M} \oplus p; \pi \leftarrow \pi'];$ 

```

Algorithm 1 is a typical primal dual algorithm: keep an upper bound (here $\pi(\mathcal{V})$) for the value of any solution (here $w(\mathcal{M})$ of any matching $\mathcal{M}$). If a solution with value equal to the upper bound cannot be found yet, we lower the upper bound carefully and continue. If $\mathcal{M}$ with $w(\mathcal{M}) = \pi(\mathcal{V})$ (inferred from CS) is found, however, $\mathcal{M}$ must have maximum weight.

3 The Hungarian Method

Even with termination, Algorithm 1 can have an exponential running time. However, one can combine PDAs with the search for augmenting paths (augpaths). For a matching $\mathcal{M}$, these are paths $p \subseteq \mathcal{E}$ such that the symmetric difference $\mathcal{M} \oplus p$ is a matching $\mathcal{M}'$ with $|\mathcal{M}'| > |\mathcal{M}|$. This guarantees termination in polynomial time. Although the principle of PD algorithms is simple, polynomial-time implementations and the verification thereof can be surprisingly hard. Because optimality criterion and path search are simpler for min-weight *perfect* matchings, we consider this problem instead.

Perfect matchings are integral solutions to $\min\{w^T x_{\mathcal{M}} \mid Ax_{\mathcal{M}} = 1, x_{\mathcal{M}} \geq 0\}$, whose dual is $\max\{1^T \pi \mid A^T \pi \leq w\}$. Feasibility of the potential (“mp-feasibility”) now means that $\pi(u) + \pi(v) \leq w(\{u, v\})$ for all $\{u, v\} \in \mathcal{E}$ *without* $\forall v \in \mathcal{V}. \pi(v) \geq 0$. WD is here that $1^T \pi \leq w^T x_{\mathcal{M}}$ for an mp-feasible π and perfect matching $\mathcal{M}$. CS is simplified to feasible $\hat{x}$ and $\hat{\pi}$ being optimum solutions for the respective LPs iff $(A^T \hat{\pi} - w)^T \hat{x} = 0$. Similarly to Lemma 4, we formalised this optimality criterion:

► **Lemma 4.** *If (1) $\mathcal{M}$ is a perfect matching in $\mathcal{E}$, (2) π is an mp-feasible vertex potential and (3) edges in $\mathcal{M}$ are tight w.r.t. π , then, $\mathcal{M}$ is a min-weight perfect matching for $\mathcal{E}$.*

We have verified executable code for max-weight bipartite matching because we reduce this and 4 other problems to min-weight perfect matching. We extend the graph $\mathcal{E}$ to a complete bipartite graph $\mathcal{E}'$ with sides of equal size ($|L'| = |R'|$ and $\mathcal{E}' = \{\{u, v\}. u \in L' \wedge v \in R'\}$), ensuring the existence of a perfect matching. If edges in $\mathcal{E}' \setminus \mathcal{E}$ should be avoided, we impose a penalty weight on them. From a min-weight perfect matching for the extended weights in the new graph, we select edges forming an optimum solution for the original problem.

Top Loop. The Hungarian Method (HM) [23, 27, 28] (Algorithm 2) expects a bipartite graph given by $\mathcal{E}, L$ and R , a weight function w and an initially feasible potential π . After checking for obvious infeasibility if $|L| \neq |R|$ and initialising $\mathcal{M}$ as $\emptyset$, the main loop repeatedly applies $\text{PATHSEARCH}()$ which returns a status flag. If $flag$ says that the dual LP is unbounded, infeasibility follows. If L is matched, the current matching is returned as min-weight perfect matching for $\mathcal{E}$. Otherwise, PATHSEARCH also returns a new feasible potential π' and an $\mathcal{M}$ -augpath p . p is used to augment $\mathcal{M}$ and the algorithm continues with the next iteration.

Algorithm 2 maintains as invariants that (HM1) $\mathcal{M}$ is a matching in $\mathcal{E}$, (HM2) $\mathcal{M} \subseteq \mathcal{E}_{\pi}$, and (HM3) π is mp-feasible w.r.t. w and $\mathcal{E}$. Provided that $\mathcal{M}$ is a matching in $\mathcal{E}_{\pi}$ and π is mp-feasible, we assume three properties for $(flag, \pi', p) = \text{PATHSEARCH}(\mathcal{E}, L, R, w, \mathcal{M}, \pi)$:

■ **Algorithm 3** *RANKING*($\mathcal{E}$ bipartite over L and R , arrival order ω for R).

```

1  $\sigma \leftarrow$  a random permutation of  $L$ ;  $\mathcal{M} \leftarrow \emptyset$ ;
2 for every arriving vertex  $u$  in  $\omega$  do
3   if  $\exists v \in \gamma(u) \setminus \bigcup \mathcal{M}$  then  $\mathcal{M} \leftarrow \mathcal{M} \cup \{\{\argmin_{v \in \gamma(u) \setminus \bigcup \mathcal{M}} \sigma(v), u\}\}$ ;
4 return  $\mathcal{M}$ ;
```

(P1) If *flag* = *dual-unbounded* there is an mp-feasible π' with $\pi'(\mathcal{V}) > B$ for any $B \in \mathbb{R}$.
(P2) If *flag* = *L-matched*, $L \subseteq \bigcup \mathcal{M}$. (P3) If *flag* = *next-iteration*, π' is mp-feasible, $\mathcal{M} \subseteq \mathcal{E}_{\pi'}$, $p \subseteq \mathcal{E}_{\pi'}$ and p is an $\mathcal{M}$ -augpath. Assuming P1-P3, the following correctness lemma holds:

► **Lemma 5.** *Let $\mathcal{E}$ be bipartite over L and R , and π be mp-feasible w.r.t. $\mathcal{E}$ and $w : \mathcal{E} \rightarrow \mathbb{R}$. On this input, the HM terminates, and returns a min-weight perfect matching if there is one.*

Path Search. We search for augpaths $p \subseteq \mathcal{E}_{\pi}$ that are tight w.r.t. an mp-feasible π , for which *alternating forests* (AF) are the central tool [17]. Other authors formalised a non-executable version for the blossom algorithm [7], and we provide an executable one that works well with forest growth and PDAs happening at the same time. An ϵ is determined similar to Algorithm 1 and preservation of mp-feasibility is similar to Lemma 2. Efficient search requires “caching” of slacks and a priority queue. We used *priority search trees* [29] for the queue and red-black trees for other data structures, resulting in a verified and executable implementation of the HM. The running time (which we did not formally verify) is $\mathcal{O}(n \cdot (n+m) \cdot \log n)$, which is the best possible running time of a purely functional implementation. The verification effort for PATHSEARCH was high with 8 involved invariants.

4 Online Matching

Online bipartite matching is a variant of bipartite matching where vertices in one party of the graph arrive online, one-by-one, each along with its incident edges. After a vertex’ arrival, the algorithm irrevocably decides on whether any of these incident edges is included in the matching. This setting was introduced by Karp et al. [25], where they considered unweighted bipartite matching. The online model of matching has recently gained substantial interest [32, 24, 21, 22] due to the proliferation of the internet and different applications which could be modelled and understood as online matching problems. Most notable among those is the Adwords [31] algorithm, which is an idealised model of how search-engine advertisers bid on keywords to show their ads to users who search for those keywords.

RANKING [25], Adwords [31], and most other online matching algorithms have proofs that are quite hard to understand, let alone formalise. Often, this is primarily due to the complex combinatorial arguments used to show them correct. For instance, RANKING’s initial analysis by Karp et al., which was formalised earlier by Abdulaziz and Madlener [6], was simplified over 6 times by a number of authors [19, 11, 13, 15, 38, 32]. Similarly, Adwords’ analysis has been repeatedly simplified to improvements to the analysis [13, 39]. In this work, as part of our framework, we formalise a PD-analysis that has substantially simplified the competitiveness analysis of a number of those online matching algorithms.

We first discuss RANKING, which works in the simplest setting, following the previous formalisation [6], and the theoretical work by Devanur et al. [13]. In its original form, RANKING operates as sketched in Algorithm 3: it takes as an input a bipartite graph with edges $\mathcal{E}$, and a list ω with the order of arrival of the online vertices R . The offline party L is

permuted uniformly randomly, leading to a list σ . We then go through the online vertices in the order given by ω , adding to the matching an edge that, if it exists, connects the incoming vertex u to the unmatched offline vertex v with the highest ranking w.r.t. σ . The neighbourhood, i.e. all vertices a vertex u is connected to, is denoted by $\gamma(u)$. The main theorem to prove about RANKING pertains to its competitive ratio:

► **Theorem 6.** *The competitive ratio of RANKING for an instance with a maximum cardinality matching of size n is at least $1 - \frac{1}{e}$, i.e. $1 - \frac{1}{e} \leq \frac{\mathbb{E}_{R \sim \text{RANKING}(G, \pi)}[R]}{n}$.*

The main idea of the primal-dual analysis of RANKING is as follows: we have a primal LP relaxation of bipartite matching $\max\{x_{\mathcal{M}} \mid Ax_{\mathcal{M}} \leq 1, x_{\mathcal{M}} \geq 0\}$ and its dual $\min\{1^T \pi \mid A^T \pi \geq 1, \pi \geq 0\}$, i.e. find π with minimum $\pi(\mathcal{V})$ s.t. $\forall u \in L, v \in R, \{u, v\} \in \mathcal{E}. \pi(u) + \pi(v) \geq 1$ and $\forall v \in \mathcal{V}. \pi(v) \geq 0$. Now, as mentioned earlier, a primal-dual analysis of an optimisation algorithm usually proceeds by proving a relationship between the primal objective and the dual objective. Proving this relation for RANKING is challenging as we need to tackle the randomisation in RANKING, which makes it a non-standard primal-dual analysis. We can only reason about the expectation of the algorithm's output and thus we can only reason about the relationship between expected values of the primal and dual LPs.

Both LPs have to be related to or derived from the output, as in the case of the two offline algorithms discussed earlier. The primal LP still corresponds to the computed matching $\mathcal{M}$, but now the main challenge is connecting the dual LP to $\mathcal{M}$: a different approach specific to the online setting is needed. In particular, the potential has to include information about the ordering of the offline vertices after they are permuted. To reason about the dual LP's expected objective, we replace the permutation step with choosing a real-numbered priority $Y_u \in [0, 1]$ for each offline vertex u . This is to allow for the expectation to be over an integrable function. Then to construct the dual solution, for each $\{u, v\} \in \mathcal{M}$, the variables $\pi(u)$ and $\pi(v)$ are instantiated by $g(Y_u)/F$ and $(1 - g(Y_u))/F$, respectively, for some $g : [0, 1] \rightarrow [0, 1]$ that is monotone and $0 \leq F \leq 1$. For all other unmatched vertices u or v , $\pi(u)$ and $\pi(v)$ are instantiated with 0. This allows us to formalise the following reasoning:

$$\begin{aligned}
 \mathbb{E}[1^T \cdot x_{\mathcal{M}}] &= \mathbb{E}[(1^T \cdot \pi) \cdot F] && \begin{array}{l} \text{(since } \pi(u) + \pi(v) = 1/F \text{ if } \{u, v\} \in \mathcal{M}, \text{ and } 0 \text{ otherwise,} \\ \text{and since the corresponding entry in } x_{\mathcal{M}} \text{ is } 1 \text{ for} \\ \{u, v\} \in \mathcal{M}, \text{ and } 0 \text{ otherwise)} \end{array} \\
 &= (1^T \cdot \mathbb{E}[\pi]) \cdot F && \text{(linearity of expectation)} \\
 &\geq 1 \cdot x^* \cdot F && \text{(by weak duality, where } x^* \text{ is an optimal primal solution)} \\
 &= 1 \cdot |\mathcal{M}^*| \cdot F && \text{(by definition of the primal LP)}
 \end{aligned}$$

for any max-card matching $\mathcal{M}^*$. The last step is choosing g and F s.t. the constructed dual solution is feasible. We set $F = 1 - 1/e$ and $g(y) = e^{y-1}$, but any assignment satisfying $\forall \theta \in [0, 1], \int_0^\theta g(x) dx + 1 - g(\theta) \geq F$ guarantees dual feasibility in expectation.¹

This approach has two advantages: it is much simpler than the combinatorial approach as formalised by Abdulaziz and Madlener as the primal-dual-based proof is less than one half (3K lines) of the combinatorial one. Moreover, it provides a general framework for reasoning about online matching algorithms. We used our development to (a) formalise the competitive analysis for the vertex-weighted variant of online matching [8], where the primal and dual LPs are adapted to include the weights with the entire analysis remaining unchanged, and

¹ The expectations are taken over the distribution of priorities Y . A step we gloss over here is the equivalence of expectations over permutations and priorities.

(b) with some changes, we did a primal-dual analysis of the Adwords algorithm [31], which solves a variant of online b-matching, which models the assignment of keywords to advertisers in the context of search engines. Interested readers should refer to the formalisations.

5 Discussion

There is a rich literature on the formal analysis of algorithms, including approximation algorithms [18], matching algorithms [7], and flow algorithms [2, 30]. Our work here, however, addresses a large gap in the literature, namely, the formalisation of primal-dual analyses of algorithms. We focus on matching algorithms, covering classical and modern results. Our formalisations, which are around 14K lines, cover a large variety of reasoning styles: the Hungarian Method (HM), which is an executable practical algorithm, and variants of online matching algorithms, which are primarily used to theoretically analyse online markets rather than to solve practical problems.

PD-based reasoning about the correctness of algorithms is mostly algebraic. The other alternative is usually a combinatorial argument, which comes with extensive case analyses and is thus harder to understand than a PD-based analysis. This pattern was noted by the many authors who worked on simplifying the analyses of online matching [19, 11, 13, 15, 38, 32].

In the formal mathematics literature, this pattern is apparent when comparing our analysis of RANKING versus the earlier one by Abdulaziz and Madlener [6]. In the combinatorial case, typically one has to refer to and characterise so-called alternating paths, which define the difference between two matchings. This is done informally via graphical reasoning (i.e. using figures), but formally this is very tedious to perform, since it e.g. incurs extensive case analyses. An example of that specific to RANKING is the function *zig-zag* characterising such an augmenting path, whose formalisation took almost 60% of the project's effort as reported by Abdulaziz and Madlener [6]. Here we do not have to characterise the difference between two matchings as such paths, and instead refer to the change in primal and dual objectives, which is formalised in a way that closely matches the informal argument.

Another step in the combinatorial proof of the RANKING algorithm (which is also common to most other maximum matching algorithms, e.g. the blossom algorithm) involves showing that the matching cannot be grown when the algorithm stops. This step uses Berge's lemma [10], whose formal proof [7] has extensive case analyses to mirror graphical reasoning about augmenting paths. Again, using the PD approach, this step is avoided for RANKING (and also HM, for instance): it is replaced by weak duality and boundedness reasoning, which is algebraic and corresponds cleanly to the pen-and-paper proofs.

In our formalisation, we have (a) weak duality and complementary slackness for different variants of LPs, and (b) lemmas to translate between graph-theoretic concepts and their encoding into matrices and vectors. In combination, we obtain the optimality criteria for the matching problems with clean proofs. We believe that this methodology is the path to follow for future formalisations of PD algorithms. Also, the common pattern should make other formalisations more understandable since problem-specific reasoning is minimised.

A big missing part in our work which we aim to contribute to a library [4] is the analysis of primal-dual approximation algorithms: these indeed form the majority of applications of the primal-dual paradigm in theoretical computer science. Our immediate plans are to formalise algorithms for approximating MaxSAT, set cover, load balancing, and Steiner trees, all of which are milestones in the theory of approximation algorithms. Another direction is minimum cost flows, for which combinatorial arguments are already formalised [2], and where the maintenance of dual variables even leads to faster algorithms [26, 35]. Also, Edmonds' weighted matching algorithm [16], which generalises the HM, is of great interest.

References

- 1 Mohammad Abdulaziz. Graph Algorithms. In *Functional Data Structures and Algorithms: A Proof Assistant Approach*. Association for Computing Machinery, 2025. doi:10.1145/3731369.3731394.
- 2 Mohammad Abdulaziz and Thomas Ammer. A Formal Analysis of Capacity Scaling Algorithms for Minimum Cost Flows. In *15th International Conference on Interactive Theorem Proving (ITP)*, 2024. doi:10.4230/LIPIcs.ITP.2024.3.
- 3 Mohammad Abdulaziz and Thomas Ammer. A Formal Analysis of Capacity Scaling Algorithms for Minimum-Cost Flows, 2026. doi:10.48550/arXiv.2602.03701.
- 4 Mohammad Abdulaziz, Thomas Ammer, Ralitsa Dardjonova, Christoph Madlener, Shriya Meenakshisundaram, and Adem Rimpapa. Isabelle Graph Library. Accessed: May 24, 2026. URL: <https://github.com/mabdula/isabelle-graph-library>.
- 5 Mohammad Abdulaziz, Thomas Ammer, Shriya Meenakshisundaram, and Adem Rimpapa. A Formal Analysis of Algorithms for Matroids and Greedoids. In *16th International Conference on Interactive Theorem Proving (ITP)*, 2025. doi:10.4230/LIPIcs.ITP.2025.2.
- 6 Mohammad Abdulaziz and Christoph Madlener. A Formal Analysis of RANKING. In *The 14th Conference on Interactive Theorem Proving (ITP)*, 2023. doi:10.4230/LIPIcs.ITP.2023.3.
- 7 Mohammad Abdulaziz and Kurt Mehlhorn. A Formal Correctness Proof of Edmonds' Blossom Shrinking Algorithm. *Journal of Automated Reasoning*, 2026. doi:10.1007/s10817-026-09747-y.
- 8 Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. Online Vertex-Weighted Bipartite Matching and Single-bid Budgeted Allocations. In *The 22nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2011. doi:10.1137/1.9781611973082.95.
- 9 Clemens Ballarin. Interpretation of Locales in Isabelle: Theories and Proof Contexts. In *Mathematical Knowledge Management*, 2006. doi:10.1007/11812289_4.
- 10 Claude Berge. Two Theorems in Graph Theory. *Proceedings of the National Academy of Sciences*, 1957. doi:10.1073/pnas.43.9.842.
- 11 Benjamin E. Birnbaum and Claire Mathieu. On-Line Bipartite Matching Made Simple. *SIGACT News*, 2008. doi:10.1145/1360443.1360462.
- 12 Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum Flow and Minimum-Cost Flow in Almost-Linear Time. In *63rd IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, 2022. doi:10.1109/FOCS54457.2022.00064.
- 13 Nikhil R. Devanur, Kamal Jain, and Robert D. Kleinberg. Randomized Primal-Dual Analysis of RANKING for Online Bipartite Matching. In *The 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2013. doi:10.1137/1.9781611973105.7.
- 14 Manuel Eberl, Johannes Hölzl, and Tobias Nipkow. A Verified Compiler for Probability Density Functions. In *Programming Languages and Systems - 24th European Symposium on Programming (ESOP)*, 2015. doi:10.1007/978-3-662-46669-8_4.
- 15 Alon Eden, Michal Feldman, Amos Fiat, and Kineret Segal. An Economics-Based Analysis of RANKING for Online Bipartite Matching. In *Symposium on Simplicity in Algorithms (SOSA)*, 2021. doi:10.1137/1.9781611976496.12.
- 16 Jack Edmonds. Maximum Matching and a Polyhedron with 0,1-Vertices. *Journal of Research of the National Bureau of Standards Section B Mathematics and Mathematical Physics*, 1965. doi:10.6028/jres.069B.013.
- 17 Jack Edmonds. Paths, Trees, and Flowers. *Canadian Journal of Mathematics*, 1965. doi:10.4153/CJM-1965-045-4.
- 18 Robin Essmann, Tobias Nipkow, Simon Robillard, and Ujkan Sulejmani. Verified Approximation Algorithms. *Logical Methods in Computer Science*, 2022. doi:10.46298/LMCS-18(1:36)2022.

- 19 Gagan Goel and Aranyak Mehta. Online Budgeted Matching in Random Input Models with Applications to Adwords. In *The 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2008. doi:10.5555/1347082.1347189.
- 20 Michel X. Goemans and David P. Williamson. The Primal-Dual Method for Approximation Algorithms and its Application to Network Design Problems. In *Approximation Algorithms for NP-hard Problems*. PWS, 1996. URL: <https://math.mit.edu/~goemans/PAPERS/book-ch4.pdf>.
- 21 Zhiyi Huang, Ning Kang, Zhihao Gavin Tang, Xiaowei Wu, Yuhao Zhang, and Xue Zhu. Fully Online Matching. *Journal of the ACM*, 2020. doi:10.1145/3390890.
- 22 Zhiyi Huang, Zhihao Gavin Tang, Xiaowei Wu, and Yuhao Zhang. Fully Online Matching II: Beating Ranking and Water-filling. In *61st IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, 2020. doi:10.1109/FOCS46700.2020.00130.
- 23 Carl Gustav Jacobi. De investigando Ordine Systematis Aequationum Differentialum vulgarium cuiuscumque. *Journal für die reine und angewandte Mathematik*, 1890. URL: https://www.lix.polytechnique.fr/~ollivier/JACOBI/Jacobi_De_Investigando.pdf.
- 24 Bala Kalyanasundaram and Kirk Pruhs. An Optimal Deterministic Algorithm for Online B-Matching. *Theoretical Computer Science*, 2000. doi:10.1016/S0304-3975(99)00140-1.
- 25 Richard Manning Karp, Umesh Virkumar Vazirani, and Vijay V Vazirani. An Optimal Algorithm for On-Line Bipartite Matching. In *The 22nd ACM Symposium on Theory of Computing (STOC)*, 1990. doi:10.1145/100216.100262.
- 26 Bernhard Korte and Jens Vygen. *Combinatorial Optimization*. Springer, 2012. doi:10.1007/978-3-642-24488-9.
- 27 Harold W. Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 1955. doi:10.1002/nav.3800020109.
- 28 Harold W. Kuhn. Variants of the Hungarian Method for Assignment Problems. *Naval Research Logistics Quarterly*, 1956. doi:10.1002/nav.3800030404.
- 29 Peter Lammich and Tobias Nipkow. Proof Pearl: Purely Functional, Simple and Efficient Priority Search Trees and Applications to Prim and Dijkstra. In *10th International Conference on Interactive Theorem Proving (ITP)*, 2019. doi:10.4230/LIPIcs.ITP.2019.23.
- 30 Peter Lammich and S. Reza Sefidgar. Formalizing Network Flow Algorithms: A Refinement Approach in Isabelle/HOL. *Journal of Automated Reasoning*, 2019. doi:10.1007/s10817-017-9442-4.
- 31 Aranyak Mehta, Amin Saberi, Umesh V. Vazirani, and Vijay V. Vazirani. AdWords and Generalized Online Matching. *Journal of the ACM*, 2007. doi:10.1145/1284320.1284321.
- 32 Milena Mihail and Thorben Tröbst. Online Matching with High Probability. In *17th International Symposium on Algorithmic Game Theory (SAGT)*, 2024. doi:10.1007/978-3-031-71033-9_2.
- 33 Julian Parsert. Linear Programming in Isabelle/HOL, 2024. doi:10.48550/arXiv.2403.19639.
- 34 Alexander Schrijver. *Theory of Linear and Integer Programming*. Wiley, 1986.
- 35 Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, 2003.
- 36 Mirko Spasić and Filip Marić. Formalization of Incremental Simplex Algorithm by Stepwise Refinement. In *FM 2012: Formal Methods*, 2012. doi:10.1007/978-3-642-32759-9_35.
- 37 René Thiemann and Akihisa Yamada. Matrices, Jordan Normal Forms, and Spectral Radius Theory. *Archive of Formal Proofs*, 2015. , Formal Proof Development. URL: https://isa-afp.org/entries/Jordan_Normal_Form.html.
- 38 Vijay V. Vazirani. Online Bipartite Matching and Adwords (Invited Talk). In *The 47th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, 2022. doi:10.4230/LIPIcs.MFCS.2022.5.
- 39 Vijay V. Vazirani. Towards a Practical, Budget-Oblivious Algorithm for the Adwords Problem under Small Bids. *Algorithmica*, 2026. doi:10.1007/s00453-026-01369-5.
- 40 Niklaus Wirth. Program Development by Stepwise Refinement. *Communications of the ACM*, 1971. doi:10.1145/362575.362577.