Learning Conference Reviewer Assignments

B. Tech. Project Stage 2 Report

Submitted in partial fulfillment of the requirements
for the degree of

Bachelor of Technology

by

Adith Swaminathan
Roll No: 06005005

under the guidance of

Prof. Soumen Chakrabarti

Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Mumbai
May 7, 2010

Contents

1 Introduction **2**
- 1.1 Conference Reviewer Assignment Problem . . . 2
- 1.2 Linear Program Formulation . . . 2

2 Previous Work **5**
- 2.1 Structured Learning for Constrained Outputs . . . 5
- 2.2 Conference Assignments as a Minimum Cost Network Flow . . . 5

3 The Minimum Cost Network Flow Framework **6**
- 3.1 Setting up the MCF . . . 6
- 3.2 Defining Edge Costs . . . 7
  - 3.2.1 The Global Loss Function . . . 8
  - 3.2.2 From Global Loss to Local Edge Costs . . . 9
- 3.3 Computing Affinities . . . 11
- 3.4 Bypassing Bimodal Behaviour . . . 12
  - 3.4.1 Penalising Reviewer Loads . . . 12
  - 3.4.2 Load Constrained Assignments . . . 13

4 Parameter Estimation **15**
- 4.1 Transductive Regression . . . 16
- 4.2 Structured SVM . . . 18

5 Results **19**

6 Future Work **21**

7 Summary **22**

Appendices **24**

A PARA : Papers Assigned to Reviewers Apparatus **24**
- A.1 Scraper . . . 25
- A.2 SearchEngine . . . 25
- A.3 PARApp . . . 26
- A.4 Wrappers . . . 26
- A.5 coREGLearner . . . 27
- A.6 SVMpythonLearner . . . 27
- A.7 CitationExtractor . . . 27

Abstract

We study the Conference Reviewer Assignment Problem, reducing it to an instance of a Minimum Cost Network Flow in an appropriate network. Building upon this formulation ([SC09]), we address the issue of estimating the parameters in the model efficiently and reliably. Two techniques are proposed, their assumptions and limitations are explored and benchmarked using a real-world dataset. Closed-form solutions are developed for some alternate formulations of the assignment problem. Finally, we describe the system that we developed to perform max-margin estimation of parameters, and thereafter conference assignments.

1 Introduction

With conferences getting larger, involving hundreds of program committee members and thousands of paper submissions, manual assignment of papers to reviewers is becoming an unmanageable task. We investigate the application of machine learning techniques to this task, namely, given some conferences' information and their assignments, we shall perform efficient parameter estimation and use this learned model to infer assignments in several other conferences.

1.1 Conference Reviewer Assignment Problem

Given a conference involving M reviewers and N submitted papers, we would like to allocate each reviewer a collection of papers to review, such that each paper is adequately reviewed. Most conferences today use bids to capture reviewer preferences; we propose features that capture a particular paper's preference over the reviewers. This is related to the growing concern that, in any conference today, the reviewer candidate's expertise (or the lack of it) in the paper's stated area is not modeled well.

The central idea in formulating this preference order over the reviewer set is the existence of a corpus of every reviewer's recent publications. With such a corpus, one can use one of several document similarity methods to detect a submitted paper's affinity to the reviewers. We shall explore this idea in 3.3.

1.2 Linear Program Formulation

Viewed abstractly, we have a set M of reviewers $\{R_i | i = 1 \dots M\}$, N submitted papers $\{P_j | j = 1 \dots N\}$. Each reviewer R_i has advertised a maximum load of L_i papers. Each paper P_j requires atleast K reviews.

> *Assumption 1 :* More sophisticated requirements can be envisioned. For eg., a paper might require atleast one review by a Systems expert and a Theory expert. Such constraints, however, make the problem of efficiently inferring assignments in our proposed model intractable.

Note that, for feasibility of any assignment, we need $\sum_{i=0}^{M} L_i \geq K \times N$. We denote the decision of assigning reviewer R_i the paper P_j with a binary decision variable y_{ij}. For the purpose of assignment, it would be useful to encode reviewer preferences for submitted papers, and paper affinities for reviewers into a global *loss*, A_{ij} for each (R_i, P_j) pair[1].With such costs, the linear program for assignment becomes,

$$\begin{aligned} \min_{y} \quad & \sum_{i=1}^{M}\sum_{j=1}^{N} A_{ij} \times y_{ij} && (1) \\ \text{subject to} \quad & \sum_{i=1}^{M} y_{ij} \geq K && ,\forall j = 1 \dots N \\ & \sum_{j=1}^{N} y_{ij} \leq L_i && ,\forall i = 1 \dots M \\ & y_{ij} \in \{0,1\} && ,\forall i = 1 \dots M; j = 1 \dots N \end{aligned}$$

For some cases, the A_{ij} can be inferred directly. For eg., in case of conflict of interest between a reviewer R_i with paper P_j, we should set A_{ij} to ∞. However, hand-tuning all A_{ij}'s directly is a bad idea for atleast two reasons.

- A very large number of these costs need to be fixed, with several consistency requirements in place. For a conference with 100 reviewers and 500 submitted papers, there are 50000 scores to be determined!
- **Multimodal Affinity Clues** There are several signals that can be exploited to generate reviewer preferences over papers and paper preferences over reviewers. However, we may not know how to differentially (de)emphasize these signals. As a first cut, some of the signals that can be exploited are :

  1. *Text Match* : One may use a textual similarity metric between submitted papers and reviewer profiles to generate TextMatch scores. Profiles can be constructed using selections of publicly available recent publications by the reviewer. Several textual similarity metrics popular in the Information Retrieval community and the Web Search community can be brought to bear. In particular, TFIDF weights with cosine similarity appear to be a viable candidate mechanism. The TFIDF weight is a basic ranking mechanism used in Information Retrieval, and encapsulates the importance of a word in a corpus to a particular document. Embedding the documents in a vector space (each dimension is a word in the vocabulary), and using cosine similarity of the TFIDF scores, one can generate similarity scores that convey the textual match between reviewer profiles and submitted papers. It is unclear, however, if TFIDF with cosine similarity is adequate or even appropriate. Alternatives to explore include

[1]One might also envision punishing certain non-assignments with an additional loss of ρ_{ij}. We shall see later that these costs can be naturally absorbed into the affinity costs, upto an additive constant.

KL divergence between suitably constructed document models (the KL divergence is a measure of *dissimilarity* of two documents. Hence, appropriate inversions of the scores are needed) and hybrids that involve document models and lexical matches.

2. *Bids* : In several conferences, a round of bidding is usually held. In previous work, we have seen these bids being used as a preliminary indicator of confidence [DN92]. We believe that bidding data, when available, can provide sharp distinctions in reviewer preferences for some papers. However, bid data is inherently sparse, which prevents us from fitting our assignments based only on bids. The bid placed by a reviewer, R_i for a paper P_j shall be denoted by b_{ij}. We shall follow the interpretations used in Microsoft's Conference Management Toolkit[2] to infer the semantics of the ordinal values of bids. Hence,

$$b_{ij} = \begin{cases} -1, & \text{if a conflict of interest has been detected between } R_i \text{ and } P_j \\ 0, & \text{if } R_i \text{ is unwilling to review } P_j \\ 1, & \text{if } R_i \text{ is indifferent to reviewing } P_j \\ 2, & \text{if } R_i \text{ is willing to review } P_j \\ 3, & \text{if } R_i \text{ is eager to review } P_j \end{cases} \tag{2}$$

3. *Topic Match* : Several conferences have various "tracks" to which a paper can be submitted; submitted papers also mention keywords that could prove useful in deciding the domain of the paper. In some conferences, reviewers are requested to highlight their areas of expertise. With this information in hand, one may design several "Topic Similarity" scores. This may be a crude 0/1 formulation (we shall denote this by t_{ij}) or more sophisticated formulations that take the track hierarchy into account.

$$t_{ij} = \begin{cases} 1, & \text{if } R_i \text{ is a reviewer in the track to which } P_j \text{ is submitted} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

4. *Citations* : Given the improving state-of-the-art in citation extraction, we can also attempt to perform citation extraction on submitted papers. One of the stronger signals we can get in this task would be if the submitted paper cites one of the reviewer's papers.

   Assumption 2 : Citations are assumed to be relevant and necessary. Even if this assumption is relaxed, we believe the presence of a citation still indicates some evidence for good reviewing capability, or atleast interest, of the cited reviewer.

   We used SeerSuite's citation extraction tool *ParsCit*[3], along with the Stanford NER parser[FGM05] to detect whether a submitted paper mentions a reviewer or not.

[2] http://cmt.research.microsoft.com/cmt/
[3] http://sourceforge.net/projects/citeseerx/

With this, we generate a 0/1 score as,

$$c_{ij} = \begin{cases} 1, & \text{if } R_i \text{ is cited by } P_j \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

As a result, the more principled approach would be to formulate the A_{ij}'s as linear combinations of these affinity clues, with the weights being tuned as required. As a general device, we can construct a feature vector of suitable dimensionalty, $\vec{\phi}(R_i, P_j) \in \mathbb{R}^d$ and train a model $\vec{w} \in \mathbb{R}^d$ such that $A_{ij} = \vec{w}^T \vec{\phi}(R_i, P_j)$.With this, the various multimodal clues can be incorporated as features in $\vec{\phi}$.
Our primary contribution is the application of structured learning techniques for assigning these weights. We have also developed an end-to-end paper assignment software that is fully compatible with CMT, and solved a few variants of the assignment problem that arise in practice.

2 Previous Work

2.1 Structured Learning for Constrained Outputs

Significant work has been done in using learning for constrained output spaces. Structured Learning has been employed successfully in Combinatorial Optimization problems to learn effective structured predictors[PRtYZ05]. Ben Taskar et al formulated very efficient linear programs that are applicable for learning one-one matchings and markov networks. Notably, this approach has applications in Statistical Machine Translation, the problem of finding optimal alignments between phrases in a sentence and its translation being one such example. The Conference Reviewer-Paper Assignment is considered as a motivating example, where the similarity between the bag of words pertaining to a reviewer and the paper is learnt through max-margin approaches[TCKG05].

2.2 Conference Assignments as a Minimum Cost Network Flow

The Conference Reviewer Assignment Problem has been addressed in many different ways; GoldSmith and Sloan have published a concise survey of various approaches used to tackle the problem. In particular, they formulate the Minimum Cost Network Flow approach that we adapt for our project[GS07]. This problem has been approached as a specific instance of a recommender system,using collaborative filtering to improve precision of the eventual assignments. This approach has been a popular one ever since the Conference Assignment problem was posed as a challenge in *IJCAI '97*. The idea of folding disparate sources of information using a factor model has been explored, with impressive improvements in assignment quality according to certain intuitive metrics[CKR09].

3 The Minimum Cost Network Flow Framework

In this section, we shall reduce the conference assignment task to an instance of the minimum cost network flow problem. Such a reduction will allow us to draw an equivalence between the Integer Linear Program stated in 1.2 (which are, in general, NP-hard) and a problem with a known polynomial time algorithm. Several polynomial time algorithms have been developed to solve the Minimum Cost Network Flow problem, [KV06] outlines a few of them. In this project, we used the Successive Shortest Path algorithm to solve MCF. We note that several other efficient algorithms for finding weighted b-matchings have been developed, the loopy belief propagation method outlined in [HJ07] being one of them.
We shall also motivate a variant of the conference assignment: Load Constrained Assignments, and show how it can be solved in closed form. Finally, the problem of distributing bid weights and computing paper-profile affinities are handled.

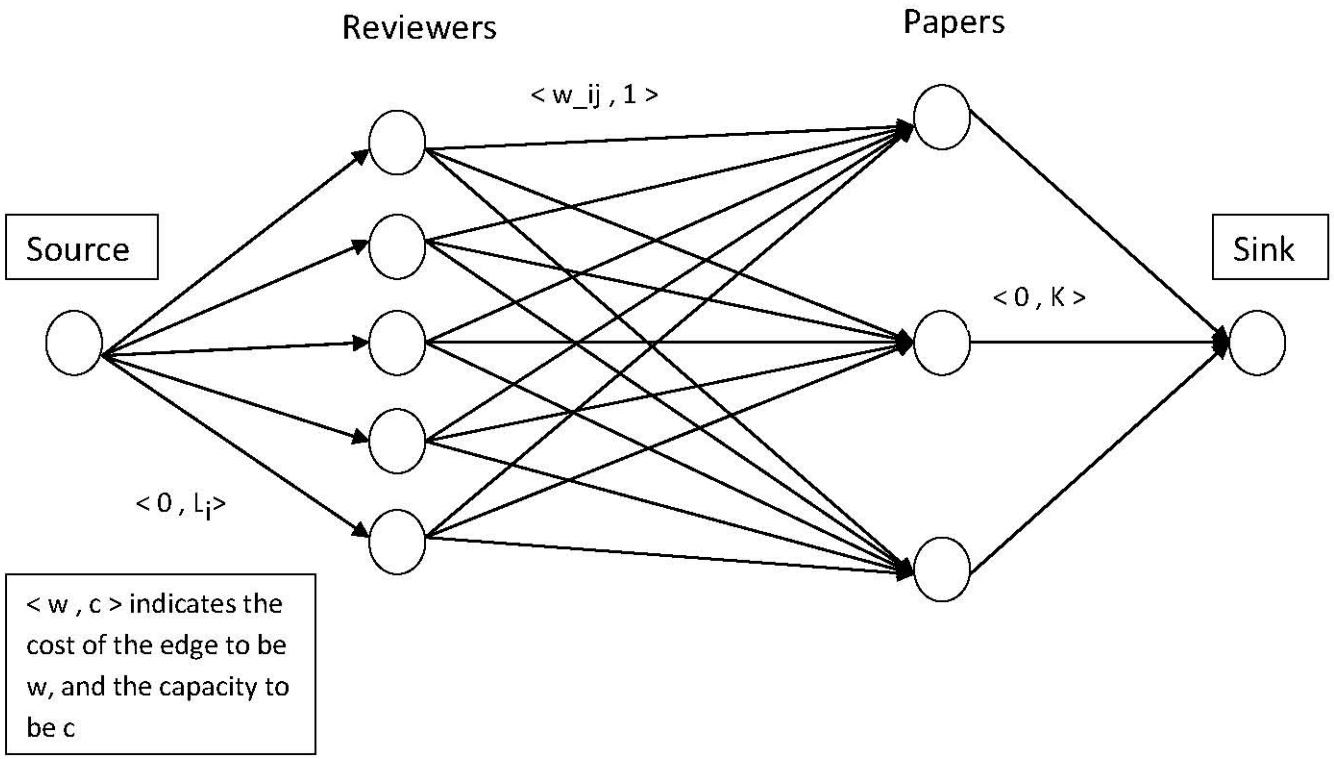

Figure 1: The Reviewer-Paper Network with a source and sink

Given a conference with M reviewers $\{R_i | i = 1 \dots M\}$, N papers $\{P_j | j = 1 \dots N\}$ and advertised reviewer loads L_i for each reviewer R_i, we shall construct a graph with $M + N + 2$ nodes

- A set of nodes $\{s_i | i = 1 \dots M\}$ with one node for each reviewer

- A set of nodes $\{t_j | j = 1 \dots N\}$ with one node for each paper
- A source 's'
- A sink 't'

Three kinds of edges are introduced

- A set of edges with capacity 1, linking each reviewer node (s_i) to each paper node (t_j); If the reviewer has a conflict of interest with that particular paper, no edge is introduced (alternatively, the capacity is made 0, or the weight is made ∞)
- A set of M edges, with capacity L_i linking 's' to each reviewer node s_i, $\forall i = 1 \dots M$
- A set of edges with capacity K linking each paper node t_j to 't'.

The cost of edges involving 's' or 't' as one of their endpoints is set to 0. The costs of the edges connecting reviewers to papers needs to be designed. If we have the scores $A_{ij} i = 1 \dots M, j = 1 \dots N$ in hand, we can assign the edge cost between s_i and t_j as A_{ij}.
A Minimum Cost Network Flow algorithm is run on this network, with the flow value set to $K \times N$. One may observe that the capacity constraints between the paper layer and the sink 't' enforces a constraint of *exactly* K reviews for each submitted paper. The solution to this set of stricter constraints can, however, be greedily extended to allow more than the minimum number of reviews for each paper. Indeed, in many conferences, this constraint is expressed as an exact equality; our flow model, hence, is equally as expressive as the linear program formulation given in 1.2.
At optimality, the flow from the reviewer layer to the paper layer can be intuitively converted into assignment decisions: if there exists flow from node s_i to node t_j at the optimal Minimum Cost Flow, we add reviewer R_i to paper P_j's list of reviewers.

3.2 Defining Edge Costs

As hinted in 1.2, we would like to combine the multimodal affinity clues into global scores A_{ij}. We shall approach this task along three fronts :

- Design an intuitive loss function, minimization of this loss function yielding assignments of superior quality (that is, minimizing the loss function is equivalent to minimizing the network flow cost, for a suitable interpretation of A_{ij}'s).
- Generate edge features that can be combined using the linear weights idea. More precisely, for each edge between reviewer node s_i and paper node t_j, we design a feature vector $\vec{\phi_{ij}}(s_i, t_j)$. The cost assigned to the edge between s_i and t_j would be related to, in our learned model, $\vec{w}^T \vec{\phi_{ij}}(s_i, t_j)$
- Generate a global feature vector over the edge features and a potential assignment, which decomposes over the edges (and hence, we can use the edge features as defined above). This global feature vector should be such that $\vec{w}^T \vec{\phi}(R \times P, \vec{y})$ yields the loss function designed above.

3.2.1 The Global Loss Function

The design of a loss function is non-trivial. Put simply, we wish to combine affinities, bids, topic overlap and conflicts in such a way that, with the resultant edge costs, if MCF were run, the cost of the optimum will indicate the *instability* of the assignment. A perceptive reader will notice a lacuna in this approach: the optimality of the assignment depends purely on the loss function. Hence, to assert that the assignment is indeed stable, we must formulate a loss function that upper bounds the *instability* in the assignment. It is unclear, however, if stability as studied in Operations Research ([GS62]) is the gold standard to aim for in this setting.

> A *stable assignment* requires that, for each reviewer-paper pair in the assignment, there does not exist more preferred executable alternatives for either the reviewer or the paper; the understanding being if there were such alternatives, the reviewer or paper have incentive to discontinue the current assignment and pursue the more lucrative alternative. However, in a conference, there is little danger of a reviewer "eloping" with a paper, ie, a reviewer revolting against his/her assigned load of papers. In place of stability, the number of bids violated promises to be a more pertinent measure. Taken to the extreme, an assignment might assert that as few bids as possible were violated (completely ignoring topics and affinities). In the other extreme, we can guide assignments purely through topics and affinities (maximising topic overlap). One of the aims of this project is to learn the right balance between these extremes.

In 3.3, we shall examine how affinity scores between reviewer profiles and submitted papers are computed. For the time being, we shall assume that the affinity scores are such that, $a_{ij} \in [0, 1]$, with a higher affinity score indicating higher similarity between the reviewer profile of R_i and the submitted paper text of P_j. For our purpose in this section, however, we need a cost-like formulation of the a_{ij}. In our experiments, we found that taking the "Affinity Loss" to be simply $(1 - a_{ij})$ was very flaky; an exponentiated loss worked better. Hence, the feature corresponding to affinity was defined as :

$$\phi_{affinity}(R_i, P_j) = \exp(-a_{ij} \cdot k_a) \tag{5}$$

where k_a is a decay constant to allow better calibration of the affinity cost. In particular, setting high values for k_a allows us to simulate a 0/1 loss for affinities; this setting of k_a was too harsh, empirically $k_a = 0.1$ worked well.
As described in **??**, we shall denote the topic match between the reviewer R_i's stated area and the paper P_j's keywords with t_{ij}; this shall, hence, be a 0/1 loss.

$$\phi_{topic}(R_i, P_j) = 1 - t_{ij} \tag{6}$$

The bid feature is a slightly less trivial case. These costs were determined based on a novel "Disappointment & Irritation" model we developed to model reviewer bids. A reviewer R_i who, due to some conflict of interest, has a bid of -1 for a particular paper P_j must not be allowed

to review the paper; the cost of such an assignment should be made ∞. Also, a reviewer R_i who expressed unwillingness to review a particular paper P_j (bid values of 0 or 1) experiences irritation on being assigned that paper, cost of such an assignment is I say: more particularly, more the unwillingness, more the irritation. Furthermore, a reviewer R_i who wished to review a particular paper P_j (bid values of 2 or 3) experiences disappointment on not being assigned that paper; let this cost be D (scaling according to the bid made). The various costs are summarised in the following *Potential* table:

With this model, the corresponding feature vector can be written down as

γ_ij	Bid = -1	Bid = 0	Bid = 1	Bid = 2	Bid = 3
0	0	0	0	D	2D
1	Infinity	2I	I	0	0

Figure 2: The Potential Table for Bid_Cost

$$\phi_{bid}(R_i, P_j, y_{ij}) = Potential(y_{ij}, b_{ij}) \tag{7}$$

We note that the above feature set can be trivially extended to incorporate citations. With scores as described in 4 in hand, we could have,

$$\phi_{citation}(R_i, P_j) = 1 - c_{ij} \tag{8}$$

With these edge features in hand, we define the global feature vector as :

$$\vec{\phi}(R \times P, \vec{y}) = \sum_{i=1}^{M} \sum_{j=1}^{N} \begin{pmatrix} y_{ij} \cdot \phi_{affinity}(R_i, P_j) \\ y_{ij} \cdot \phi_{topic}(R_i, P_j) \\ \phi_{bid}(R_i, P_j, y_{ij}) \\ y_{ij} \cdot \phi_{citation}(R_i, P_j) \end{pmatrix} \tag{9}$$

3.2.2 From Global Loss to Local Edge Costs

We now need to formulate edge costs between the reviewer layer and paper layer such that a flow through the network will have cost corresponding to the loss function for that particular assignment. As a pre-emptive measure, we set all edges with corresponding bid of -1 (conflict) to have an infinite cost. In the rest of this exposition, bids of -1 (and the corresponding edges) are ignored.

We observe that the global loss equals $\vec{w}^T \vec{\phi}(R \times P, \vec{y})$. From 9, we have,

$$Global\ Loss = \sum_{i=1}^{M} \sum_{j=1}^{N} w_1 \cdot y_{ij} \cdot \phi_{affinity}(R_i, P_j) + w_2 \cdot y_{ij} \cdot \phi_{topic}(R_i, P_j) + w_3 \cdot \phi_{bid}(R_i, P_j, y_{ij}) \tag{10}$$

We observe that the first two terms in the loss are dependant only on the assignment variables y_{ij} that are set to 1. This can be easily converted into the MCF formulation: to the edge

connecting s_i and t_j, simply assign $w_1 \cdot \phi_{affinity}(R_i, P_j) + w_2 \cdot \phi_{topic}(R_i, P_j)$ as the cost of that edge. We shall refer to the third term as Bid_Cost, and this can be manipulated to yield,

$$\begin{aligned} \phi_{bid}(R_i, P_j, y_{ij}) &= Potential(y_{ij}, b_i j) \\ &= y_{ij} \times \{(I \cdot [[Bid == 1?]]) + (2 \times I \cdot [[Bid == 0?]])\} \\ &+ (1 - y_{ij}) \times \{(D \cdot [[Bid == 2?]]) + (2 \times D \cdot [[Bid == 3?]])\} \end{aligned} \tag{11}$$

Summing this cost across all reviewers capable of reviewing the paper (ie, non-conflicting reviewers), we find that the *assignment independant bid cost*, C_{P_j} is,

$$C_{P_j} = N_+(P_j) \times D + 2 \times N_{++}(P_j) \times D \tag{12}$$

where N_+ is the number of +2 bids for that paper, N_{++} is the number of +3 bids for that paper P_j. For any feasible assignment, we know that paper P_j receives K reviews. Hence, we add a cost of $\frac{C_{P_j}}{K}$ to each edge that corresponds to a non-conflicting author. The remaining component of the Bid_Cost is dependant purely on the assignment variable, and can be added similar to the first two terms.

Determining Disappointment An important fact that we glossed over in the development of the previous section is the complex interplay between the first two terms in the global loss and Bid_Cost. For ease of inference, we would like this interrelationship to be limited. In particular, we note that the first two terms are bounded by $w_1 + w_2$. We propose an intuitive motivation for the concept of disappointment which ensures that the Bid_Cost remains bounded.
A key observation is that disappointment on violation of a positive bid cannot be independant of the paper P_j. In particular, if a paper is known to be a "favourite", one cannot be too disappointed if one bid for it and did not get it. In a sense, the collective disappointment of the set of interested reviewers for a particular paper is conserved. Formally, we have used the following formula to set disappointment.

$$D = \frac{K}{(N_+ + 2 \cdot N_{++})} \tag{13}$$

With this formula for disappointment, we note that the Bid_Cost will be bounded from above by 2, and from below by $1-K$. Given that, in most conferences, K is ~ 3 *to* 5, these bounds are tight enough to not impact interpretation. A more realistic way of assigning disappointment might be on a *per-reviewer* basis, or by using a more complicated monotonically decreasing function of the number of positive bids. However, the utility of such a definition of bid disappointment is unclear and asserting bounds on Bid_Cost convoluted.
At this point, given a vector $\vec{w}$ (which shall be the learned model), we have constructed an instance of Minimum Cost Flow, such that running a MCF solver on this graph is equivalent to minimizing the linear program formulated in 1.2.

3.3 Computing Affinities

An important subproblem in computing the edge features is that of calculating the *affinity* between a reviewer and a paper. We approach this problem by breaking it into two sub-tasks, generating reviewer profiles and computing similarity between profiles and papers.

Generating Reviewer Profiles One might assume that, given a reviewer's homepage, a profile of his/her recent publications can be constructed. However, the profile construction problem has an asymmetric response to the documents collected- a single irrelevant document could spoil the entire profile, while every additional relevant document marginally improves the quality of the profile. In technical terms, we need very high precision (even at the expense of recall). Inspite of elaborate heuristics, this approach routinely collected documents detailing a semester time-table, or a poem, etc.
Our current approach uses DBLP[4] to reliably infer the titles of recent publications of a reviewer, followed by Google Scholar[5] queries to actually locate the available documents. We believe this is an ideal balance between precision- enforced by DBLP, and recall- facilitated by Google Scholar. To compute the affinities, we wanted to use a bag-of-words approach which requires high fidelity preservation of the words in the document. The quality of the profiles constructed will crucially determine how good a signal the affinities will turn out to be.
Several alternatives can be used to assign a similarity score between the set of papers for a reviewer and a particular conference paper. The probabilistically grounded approach of building a document model for each paper in a reviewer's profile, one for each of the conference papers, and taking a suitable similarity measure between the models suffers from the complication of a non-trivial recombination phase: given twenty such scores, how to combine them into a single score signifying similarity between the entire profile and the paper? Such a recombination can be bypassed by using "SuperDocuments" for each reviewer (which is simply the concatenation of all collected papers for that reviewer). However, a document model built from this SuperDocument becomes less descriptive as the number of duplicated documents collected for that reviewer grows. The best approach appears to be clustering the input documents, and using the "mean" documents in the SuperDocument generation phase. A promising similarity metric in this context appears to be the Kullback-Leibler Divergence.

> *Kullback-Leibler divergence* is an assymetric dissimilarity measure between two probability distributions. It is defined as $D_{kl}(P||Q) = \sum_i P(i) \cdot \frac{\log P(i)}{Q(i)}$ for distributions P and Q over discrete variables.

We settled on a different approach for PARA, using this intuition of collating all documents of a profile into a single "SuperDocument". In essence, we wished to exploit the term frequency/inverse document frequency of words appearing in these documents. A Lucene index[6]

[4] http://dblp.uni-trier.de/db/about/faqdblp.html
[5] http://scholar.google.co.in/intl/en/scholar/about.html
[6] http://lucene.apache.org/java/docs/

was built over this corpus of SuperDocuments, using a custom analyzer. Each conference paper's contents were also passed through this custom analyzer and, using these contents as a search term, the index search was performed, hits collected and scored. As a first cut, we have used the Lucene Index Search scores to be indicative of the affinity between a particular reviewer and a particular conference paper.

3.4 Bypassing Bimodal Behaviour

Although the results proved in earlier sections are theoretically sound, in practice we found severe shortcomings in the linear program in the first place. In a test conference with 300 reviewers, each assumed to have a maximum advertised load of 15 papers, we found that reviewers either had very few assigned papers, or were assigned the maximum permissible number of papers. Such bimodal behaviour has also been observed in machine translation, where introducing "fertility" of words into a language model yields either promiscuous words or unattached words[LJTKJ06]. The core of the problem lies in the fact that the cost of an assignment is assumed to be independant of other assignments (bid costs model a weak interdependancy, but this is only for the positive bids, and at a very coarse level). Indeed, such an assumption is essential if the problem is to remain tractable. However, we can model certain inter-assignment dependancy (over and above that captured by the minimum cost flow) using ideas of load penalties and rebalancing penalties. We outline two particular solutions to tackle the problem :

- Reformulate the Linear Program, introducing several new parameters which model increasing load penalties for a reviewer. These parameters will have additional inter-parameter constraints and will need to be learnt.
- Rebalance the loads given a set of (assumed optimal) assignments using a modified Minimum Cost Flow instance

3.4.1 Penalising Reviewer Loads

A survey of our formulation in 1.2 indicates why the bimodal behaviour is present : there is no incentive in the model for a balanced load across all reviewers. We wish to incorporate a mechanism that promotes low loads across all reviewers, while allowing higher loads on a per-paper basis as required. This can be achieved by penalising higher loads, the penalty being decided by the features of the reviewer paper pair $\langle R_i, P_j \rangle$.
To handle varying loads for each reviewer, we must introduce new indicator variables (rather than simply y_{ij}). Let y_i^d indicate that reviewer R_i has a load greater than or equal to d. Let the penalty for increasing a reviewer's load from $d-1$ to d be w_i^d. The total penalty for a reviewer R_i is,

$$\delta_i = \sum_{d=1}^{L_i} w_i^d \cdot y_i^d \tag{14}$$

The transformed linear program becomes,

$$\min_{y} \quad \sum_{i=1}^{M}\sum_{j=1}^{N} A_{ij} \cdot y_{ij} + \sum_{i=1}^{M}\sum_{j=1}^{L_i} w_i^d \cdot y_i^d \tag{15}$$

$$\text{subject to} \quad \sum_{i=1}^{M} y_{ij} \geq K \qquad ,\forall j = 1 \dots N$$

$$\sum_{j=1}^{N} y_{ij} \leq \sum_{d=1}^{L_i} y_i^d \qquad ,\forall i = 1 \dots M$$

$$y_{ij} \in \{0,1\} \quad ,\forall i = 1 \dots M; j = 1 \dots N$$

$$y_i^d \in \{0,1\} \quad ,\forall i = 1 \dots M; d = 1 \dots L_i$$

This program also has an equivalent Minimum Cost Network Flow instance. We note that in the construction given in 3.1, we attached the source 's' to each reviewer node s_i, with capacity L_i, and cost 0. This edge shall be "split up" into L_i edges, each with capacity 1, and cost of the d^{th} edge in the set being w_i^d.
The issue of parameter estimation for this problem, however, is harder since there are several orders of magnitude more parameters to fit. As a solution, we might attempt to reduce the number of parameters by picking a family of parametrised concave functions, with parameters α. These concave functions should be such that they attain the value 0 at 0, and ∞ at a value $\geq L_i$. We note that fitting the additional parameters w_i^d to points on these functions will give us the desired solutions, with probably fewer parameters to estimate (namely, the α's) . We do not use this approach in the project, primarily due to the complexity of the parameter estimation step.

3.4.2 Load Constrained Assignments

An alternate approach to ensure lower variance in reviewer loads is to perform a "rebalancing" step given some assignments. Note that the MCF framework elucidated in 3.1 allows us to fix some partial assignments. In particular, if we are given a fixed assignment between reviewer R_i and P_j, we simply remove the edge between s_i and t_j in the graph, and enforce the paper in-degree constraint to be $K - 1$, for the paper P_j. However, running MCF on this modified graph will still exhibit bimodal behaviour.
Consider an assignment $\vec{\gamma}$, such that the per-paper constraints are satisfied, though the per-reviewer load constraints may be violated. In particular, $\vec{\gamma}$ is such that,

$$\sum_{i=1}^{M} \vec{\gamma}_{ij} = K \qquad ,\forall j = 1 \dots N$$

$$\vec{\gamma}_{ij} \in \{0,1\} \quad ,\forall i = 1 \dots M; j = 1 \dots N$$

With such a $\vec{\gamma}$ in hand, we would like to find a "rebalanced" assignment that satisfies load constraints L_i for each reviewer R_i and which also tampers with a low number of the given assignments $\vec{\gamma}$. The two extremes that are possible are :

- The given assignments are ignored completely. MCF is run with the load constraints L_i for each reviewer R_i in place. The output is expected to be bimodal.
- The given assignments are pruned until they satisfy the load constraints. In particular, every overloaded reviewer drops his least preferred papers, whcih then get reassigned according to the MCF output.

The second approach, though appealing, is sub-optimal. Ideally, for the collective good, a paper that has a good replacement reviewer is the one which should be dropped from an overloaded reviewer's reading list. The tradeoff between these two extremes shall be modeled by a parameter, ϵ. More precisely, the cost of tampering with a given assignment is assumed to be $\frac{\epsilon}{2}$ [The structure of $\vec{\gamma}$ ensures that changes to the assignment must occur in pairs]. With this notation, we have a modified linear program :

$$\min_{y} \quad \sum_{i=1}^{M}\sum_{j=1}^{N} A_{ij} \cdot y_{ij} + \frac{\epsilon}{2} \cdot [[y_{ij} \neq \gamma_{ij}]] \tag{16}$$

$$\begin{aligned} \text{subject to} \quad & \sum_{i=1}^{M} y_{ij} = K && ,\forall j = 1 \dots N \\ & \sum_{j=1}^{N} y_{ij} \leq L_i && ,\forall i = 1 \dots M \\ & y_{ij} \in \{0,1\} && ,\forall i = 1 \dots M; j = 1 \dots N \end{aligned}$$

An equivalent Minimum Cost Flow instance can be set up as follows : The constructed graph shall have $M + N + 2$ nodes (in this regard, the nodes are the same as the original MCF formulation) :

- A set of nodes $\{s_i | i = 1 \dots M\}$ with one node for each reviewer
- A set of nodes $\{t_j | j = 1 \dots N\}$ with one node for each paper
- A source 's'
- A sink 't'

Four kinds of edges are introduced

- A set of M edges with capacity L_i linking each reviewer node s_i to 't'. These edges shall prevent any reviewer from being overloaded.
- A set of M edges with capacity $\sum_{j=1}^{N} \vec{\gamma}_{ij}$ linking the source 's' to each reviewer node s_i. These edges represent the load of the reviewer under the $\vec{\gamma}$ assignment.
- A set of $K \times N$ edges with capacity 1, linking a reviewer node (s_i) to each paper node (t_j); If the reviewer has been assigned that particular paper under the $\vec{\gamma}$ assignment, this edge is added to the graph.

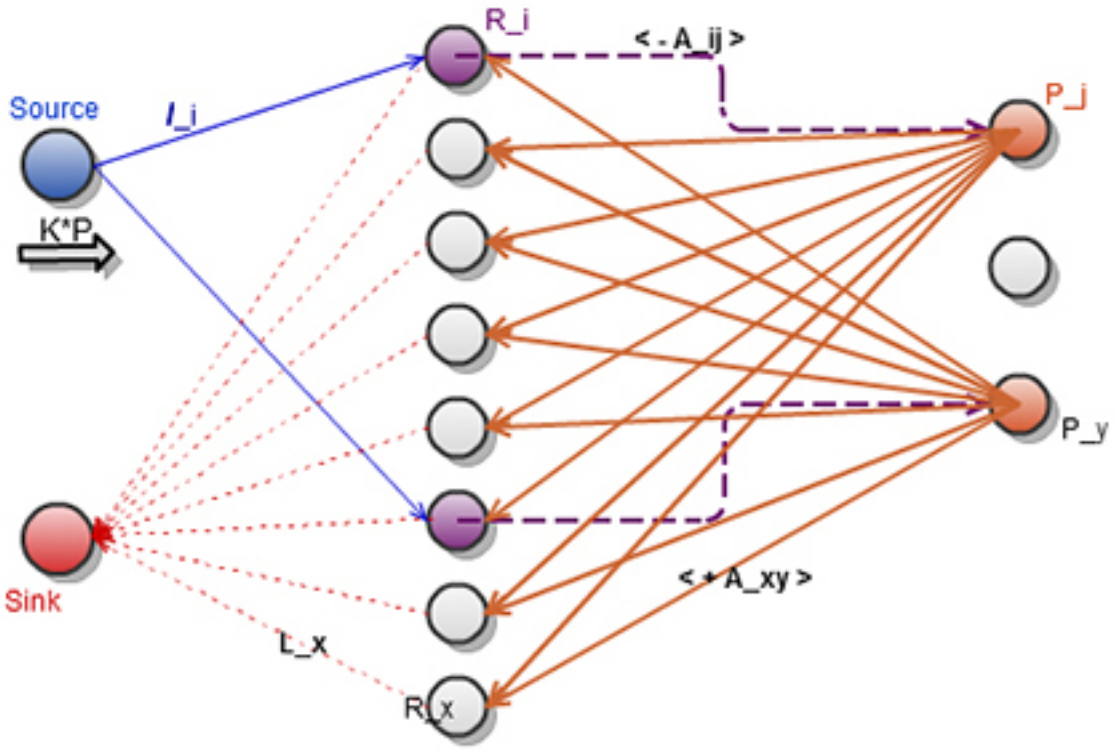

Figure 3: The Load Constrained Assignment MCF Instance

- A set of $M \times N$ edges, with capacity 1 linking each paper node (t_j) to each reviewer node (s_i), $\forall i = 1 \ldots M; j = 1 \ldots N$.

The cost of edges involving 's' or 't' as one of their endpoints is set to 0. The costs of the edges connecting reviewers to papers and vice versa needs to be designed. We note that the edge costs assigned in the MCF formulation given in 3.1 is independant of the load constraints. Hence, these edge costs can be precomputed for every reviewer paper pair, we shall call these precomputed scores A_{ij}. To an edge linking a reviewer node s_i to t_j, we assign a cost of :

$$Cost_{s_i \to t_j} = \frac{\epsilon}{2} + A_{ij} \tag{17}$$

whereas, for the edge linking paper node t_j to node s_i, we assign :

$$Cost_{t_j \to s_i} = \frac{\epsilon}{2} - A_{ij} \tag{18}$$

A Minimum Cost Network Flow algorithm is run on this network, with the flow value set to $K \times N$. It is easy to see that this network encapsulates the linear program set out in 16.

4 Parameter Estimation

We now turn to the issue of efficient parameter estimation. This task is motivated by the concern that we may not know how to differentially (de)emphasize various multimodal signals, derived

from various domains and having various scales. The issue of efficient parameter learning in structured output spaces has been well studied, and several decomposition based approaches and exponentiated loss approaches have shown promising results in practice.
For this task, we shall assume that there are several conferences available as training data. Each conference is a tuple of the form $\langle R, P, B, \vec{y} \rangle$, R being the set of reviewers, P the set of submitted papers, B the bids placed by reviewers for papers (this includes conflicts of interest modelled as special bids of -1), and $\vec{y}$ the eventual assignments performed. Note that, given the reviewer set R and paper set P, we can use the techniques developed in 3.3 to generate affinity scores, and the equation defined by 3 to decide the topic match scores. We can often get more expressive outputs rather than the boolean output vector $\vec{y}$. In several conferences, after the first round of reviews, each review is accompanied by a reviewer confidence. We have seen previous work where bids were assumed to be a crude presumption of eventual reviewer confidence. We motivate more principled ways to integrate reviewer confidence into the parameter estimation algorithm to yield better results.
We outline two techniques that were attempted in this project :

- *Transductive Regression* : The task of estimating parameters is reduced to one of ordinal regression, where each activated assignment in each input conference is assumed to be in the training data. The parameters once learned, are used to guide assignments in the test conferences.

- *Structured SVM* : We outline a Support Vector Machine formulation of the problem, which is known to yield efficient approximate solutions.

Each of the above approaches has certain drawbacks that are described in their respective subsections. These techniques are a proof of concept that efficient parameter estimation can be performed in this setting; the best method to do it, however, is still unclear.

4.1 Transductive Regression

For this approach, it will be useful to view the reviewer confidence information as a matrix, $\mathbb{C}$ of dimension $M \times N$. Several values of this matrix are "hidden" by default. This is because, of the possible $M \times N$ cells in $\mathbb{C}$, only $K \times N$ of these are observed (ie, only these many assignments are carried out, which will have reviewer confidence information accompanying them). And, in a typical conference, $K \ll M$. This observation thwarts any approach that were to estimate parameters based on the observed data alone.
We wish to generate augmented training sets such that performing naive parameter estimation on the training set could yield "good" values for the parameters in the test conferenes. As pointed above, restricting the training set to have only the $K \times M$ observed edges would be too sparse. We propose a three step alternative :

1. Heuristically expand the set of observed edges to also include *probable alternate* edges; these edges are the unassigned edges that could have been observed if not for the load

constraints. We note that the original edges in the set have reviewer confidence lables while the heuristically added ones do not.

2. Viewing this expanded set of edges as an instance of transductive learning, we learn a regressor that assigns a label to all the unlabelled edges in the set.

3. With this expanded set of edges, each with labels, we perform naive parameter estimation assuming each edge was observed independantly. In particular, a logistic regressor is used to fit the parameters.

Augmenting Training Data The heuristic we used to augment the given training data was derived using three assumptions.

- A non-indifferent bid expressed by a reviewer R_i for a paper P_j expresses (dis)interest, and is hence a viable indicator of confidence.
- A strong topic match between a reviewer's stated area and the paper's keywords indicates that the reviewer could have been a good reviewer for that paper. This is probably a positive indicator of reviewer confidence.
- All other edges are essentially inconsequential to the task of determining reviewer confidence. These edges, hence, will not belong to the augmented set.

We call this augmented training dataset $\mathbb{D}$.

Transductive Labelling Given such an augmented set of edge feature vectors, we need to find labels for the unlabelled edges. This is an instance of transductive learning. In particular, since the outputs (reviewer confidence) are ordinal in nature, we can view this as an instance of transductive regression.
We use a co-training algorithm to find the labels for the unlabeled set. This method was developed in [ZL07], and is indicative of the popular k-NN approach used for this problem. We embed the edge feature vectors in an appropriate vector space, and use two $k - NearestNeighbours$ classifiers (each deciding the nearest neighbours with a different metric) that complement each other. This means, one classifier picks the next example which is to be labelled by the other classifier, and vice versa. For our experiments, we used the Euclidean distance as the distance metric for one classifier, and used the Mahalanobis Distance[7] as the metric for the other. After this step, we are assured that each element $d \in \mathbb{D}$ is a labelled training example.

Logistic Regression At this point, we have an augmented training set $\mathbb{D}$ which contains elements of the form $\langle \begin{pmatrix} x_i^1 \\ x_i^2 \\ x_i^3 \end{pmatrix}, y_i \rangle \in \mathbb{D}$. We use logistic ordinal regression on this dataset to find

[7] $D_M(\vec{x}, \vec{y}) = \sqrt{(\vec{x} - \vec{y})S^{-1}(\vec{x} - \vec{y})}$, where S is covariance

the best fit for the parameters. For convenience, we shall define the logistic loss function as,

$$h(z) = \ln(1 + \exp(z)) \tag{19}$$

To perform ordinal regression, we shall divide the real number line into segments, each segment corresponding to an ordinal attribute value of the reviewer confidence. If $\{\theta_0, \theta_1, \theta_2 \ldots \theta_k\}$ are the segment parameters, we can write down the regression program as,

$$\hat{w} \quad = \quad \arg\max_{w} \sum_{i \in \mathbb{D}} h(\theta_{y_i - 1} - \vec{w}^T \vec{x_i}) + h(\vec{w}^T \vec{x_i} - \theta_{y_i}) + \frac{\lambda}{2} \cdot \vec{w}^T \vec{w} \tag{20}$$

With the learned parameters $\hat{w}$, we can plug them into the inference engine for any test conferences to yield assignments. We used the Zelig system developed in [KIL07], and have reported the results in 5.
Note that this approach does not exploit the structure in $\vec{y}$ and makes some rather crude assumptions about reviewer confidence and assignments. Specifically, there is no clear justification (other than an intuitive need to eliminate overwhelming bias for a particular confidence ordinal value) for the three heuristics we listed above while augmenting our training dataset.

4.2 Structured SVM

In this section, we want to explore the efficacy of using the structure of $\vec{y}$ to perform better parameter estimation. Our input is a set of training instances (each instance being a conference), $\langle R^i, P^i, B^i, \vec{y^i}, \mathbb{C}^i \rangle_{i=1}^{c}$, R^i being the set of reviewers in the i^{th} conference, P^i the set of papers submitted in that conference, B^i the bids placed by reviewers for papers, and $\vec{y^i}$ the eventual assignments performed, the final term in the tuple is the confidence matrix as defined in 4.1. The c is the total number of conferences in our training set. To allow an efficient Support Vector Machine formulation of this task, we shall change the feature vectors slightly,

$$\begin{aligned} \vec{\phi}_{SVM}(R^i \times P^i, \vec{y^i}) &= -\vec{\phi}(R^i \times P^i, \vec{y^i}) \\ &= \sum_{k=1}^{M^i} \sum_{j=1}^{N^i} \begin{pmatrix} -{y^i}_{kj} \cdot \phi_{affinity}({R^i}_k, {P^i}_j) \\ -{y^i}_{kj} \cdot \phi_{topic}({R^i}_k, {P^i}_j) \\ -\phi_{bid}({R^i}_k, {P^i}_j, {y^i}_{kj}) \\ -{y^i}_{kj} \cdot \phi_{citation}({R^i}_k, {P^i}_j) \end{pmatrix} \end{aligned} \tag{21}$$

This allows us to write the original linear program as a maximization problem rather than a minimization problem.
Intuitively, we would like to find parameters $\vec{w}$ such that they can at least predict the correct assignments in the training data. Namely,

$$\vec{y^i} = \arg\max_{\vec{\gamma} \in \Upsilon^i} \vec{\gamma}^T \vec{\phi}_{SVM}(R^i \times P^i, \vec{y^i}) \tag{22}$$

for each i=1 ... c.

Here, Υ is exponential in the size of $M^i \times N^i$ and denotes all possible assignments. We shall define a "Confidence-weighted Loss" function that denotes the dissimilarity between two possible assignments. The "Confidence-weighted Loss", $\Delta_{\mathbb{C}}(\vec{\gamma}, \vec{y})$ is defined by :

$$\Delta_{\mathbb{C}}(\vec{\gamma}, \vec{y}) = \sum_{i=1}^{M} \sum_{j=1}^{N} [[\gamma_{ij} \neq y_{ij}]] \cdot \mathbb{C}_{ij} \tag{23}$$

With the *Confidence-weighted loss* and the feature vectors $\vec{\phi}_{SVM}$ in hand, we can define a Support Vector Machine as follows.

$$\begin{aligned} \min_{\vec{w}, \vec{\xi}} \quad & \frac{||\vec{w}||^2}{2} + \frac{\Gamma}{c} \cdot \sum_{i=1}^{c} \xi^i \\ & \text{subject to} \\ \vec{w}^T \cdot \vec{\phi}_{SVM}(R^i \times P^i, \vec{y^i}) - \vec{w}^T \cdot \vec{\phi}_{SVM}(R^i \times P^i, \vec{\gamma}) \geq \Delta_{\mathbb{C}^i} - \xi^i \quad & , \forall \gamma \neq \vec{y^i} \\ & \forall i = 1 \ldots c \end{aligned} \tag{24}$$

where Γ is the regularization parameter. This program has an equivalent form,

$$\min_{\vec{w}^T \vec{w} \leq \Gamma} \quad \sum_{i=1}^{c} \max_{\vec{\gamma} \in \Upsilon^i} \{\vec{w}^T \cdot \vec{\phi}_{SVM}(R^i \times P^i, gam\vec{m}a) + \Delta_{\mathbb{C}^i}(\vec{\gamma}, \vec{y^i})\} - \vec{w}^T \cdot \vec{\phi}_{SVM}(R^i \times P^i, \vec{y^i}) \tag{25}$$

Note that the term inside the cascaded max is the term we would have to compute for inference alongwith an additional term signifying loss. We saw how inference could be done in polynomial time by using the Minimum Cost Flow framework; we note that the loss function is decomposable over the edges. As a result, we can simply add the appropriate edge cost to each edge in the Minimum Cost Flow Network and generate a solution for the *loss augmented inference* problem. The cutting plane algorithm formulated by [THJA04] gives efficient approximate results for this quadratic program.

5 Results

After a round of reviews, several conferences have reviews annotated with reviewer confidence. We believe this is a reliable indicator of the quality of that assignment. The problem, however, is that the confidence data is extremely sparse. Inability to extrapolate the observed confidence scores to every possible reviewer-paper pair, and the ineffectiveness of assuming a default confidence score for any unobserved assignment forces us to look for alternative measures of assignment quality. We propose the following intuitive metrics to decide the optimality of an assignment :

- *Fraction of Positive Bids violated.* We shall denote this by #(+)
- *Fraction of Negative Bids violated.* This shall be denoted by #(-)
- *Fraction of Topic Matches satisfied.* This shall be denoted by #(T)

Parameters	CoREG(Normal)	CoREG(Ologit)	SVMstruct
Affinity_Param	0.1547	0.2482	0.0
Bid_Param	0.2936	3.5782	1.0022
Topic_Param	0.2766	2.2347	0.0823
Citation_Param	0.1989	1.7279	1.4913

Table 1: Parameters Learned

Metric	CoREG(Normal)	CoREG(Ologit)	SVMstruct	WWW2010
#(+)	0.5347	0.5201	0.4826	0.5850
#(-)	0.0	0.0	0.0	0.0206
#(T)	0.9201	0.8959	0.8039	0.7707

Table 2: Performance

Even with these objective metrics, there are issues that prevent us from using these to benchmark various assignment algorithms. The concept of a "Positive" and "Negative" bid is open to interpretation : Eg., CMT views only bids of 0 (Unwilling) as a negative bid, whereas we treat bids of 1 (In a pinch) as a mildly negative bid. It is not clear if these metrics together are representative of quality, and if they are, what combination of these metrics should one optimise for.

We collected reviewer profiles for the WWW2010 Conference, and assisted in the assignment process for this conference. We learned model parameters using each of the approaches, and inferred assignments on the same dataset. The results of this are shown below, and contrasted with the values obtained for the actual assignment.

We have also generated reviewer profiles for WSDM2008, however, this dataset is currently incomplete without the fulltext of the submitted papers (performing the experiment on only the accepted papers is biased). We hope to complete this task in the weeks to come.

The transductive regression approach allows several variations where one might use one of many possible regression models. We picked two models : the normal and the ordinal logistic model as representatives. The results in indicate that better design choices can be made. For example, the very low weights assigned to affinities by all models indicate that, perhaps, the vector space cosine similarity is not the ideal similarity metric for this task. Noise in the collected profiles is a major concern, and better profile quality needs to be supplemented by appropriate similarity metrics for the profile-paper affinities to be exploited fully. The comparatively high weights assigned to bid potentials validate the assumption that bids are a precursor of confidence. As was expected, citations have a non-trivial weight; we believe that better citation matches can provide significant improvement in the quality of this signal. The relatively high weights given to topic matches in the CoREG approach also makes sense, since the training data was biased by heuristically adding only those reviewer-paper pairs that had a topic match.

5 shows that both a trained model can be competitive with manual assignments (evaluated

on these metrics). As noted above, the definition of a negative bid could cause some biases in the results. Generalisability and the applicability of these metrics remain major concerns.

6 Future Work

One subtask of the current work which is expected to have a high impact on the conference assignment problem is that of profile generation: Given an author's name, can we reliably generate a profile of his/her most recent publications? The idea of clustering collected documents to glean the *areas of interest* of the author promises to be an effective solution to the associated problems of duplication (the same paper collected from many different sources) and outlier papers(an irrelevant paper). Our current approach leverages DBLP and Google Scholar; once a suitable mechanism to assess the quality of a profile is in place, we believe the merits of various different approaches can be quickly judged.

Another closely related issue is the mechanism to compute similarity between profiles and papers. We have used a TF-IDF score computed by embedding the profile and the paper in a suitable vector space; it is unclear if this approach is necessarily the optimal choice in the current setting. Alternate approaches to compute the profile-paper similarity could be a document-model based approach, with Dirichlet priors over the set of topics available in the conference.

One important signal that we overlook currently is that of transitive citations. We believe merely looking at "Paper Cites Reciewer" citation information is too noisy a signal to be a reliable indicator of Reviewer expertise. For example, some papers could have several references to other papers in areas not central to the theme of the paper (this report is a case in point). We can alleviate the problem somewhat by mining more extended citation relations, say, how often the author of the submitted paper is cited by the reviewer's profile, etc.

As we saw in 3.4.1, parameter estimation for the load-penalty linear program is a non-trivial task. The two approaches we have used each have certain shortcomings :

- The transductive regressor does not account for the rich inter-dependancy between different edges. We attempted to heuristically overcome this problem, but it is an unprincipled solution at best.
- The structured SVM approach uses the cutting plane algorithm, and minimizes an upper bound to the objective, rather than the objective itself. Approximation bounds have been provided but an exact solution is highly desirable.

Currently, we need a subjective analysis of the output assignments. We saw some preliminary approaches in using reviewer confidence as a measure of assignment quality. An alternate approach could be to insist on stability, model the conference as a co-operative game, and pick an assignment in the core of the game. However, such an approach does not lend itself to machine learning techniques at first glance.

A fundamental assumption in our entire formulation has been that a paper is indifferent to the reviews it gets (hence, we could simply assert a particular constant flow into a paper node in

the MCF formulation). More elaborate requirements can be envisioned, which model current conference assignments better. In particular, one might insist that a paper's theory, application and system be adequately reviewed. Each reviewer could be classified as an expert in one of these fields. Alternatively, one might insist that assignment quality depends on it's popularity - if each reviewer gets at least one paper he/she bid for, the assignment is popular. Each of these alternate formulations require significant revisions to our formulation, and are viable areas for future work.

7 Summary

The primary focus of this project is to investigate the efficacy of structured learning techniques in the domain of Conference Reviewer-Paper assignments. In this regard, we used max-margin techniques as developed in [TCKG05]. Several interesting problems have also been discovered during the course of the project.

We cast the problem as a specific instance of a Minimum Cost Network Flow, and proposed an intuitive loss function . Minimising the cost of the network flow was shown to be equivalent to optimising the loss function. We then addressed the issue of model parameters and used structured learning techniques to fit these parameters to data. In particular, we saw two approaches and their results on a specific dataset. The first approach assumed that each individual assignment was independant of the other, and heuristically, the training dataset was expanded to allow for some trivial dependancies in the input. The second approach used support vector machines in structured output spaces, but used the cutting plane algorithm which gave only an approximate optimum.

We must acknowledge that any human need in a mining process is subjective end to end, so a collection of intuitive measures is all we have to guide the training process. Sparsity of input data is another complicating factor, making the feedback process more tenuous. With these challenges in mind, we attempted to tackle a well motivated problem. We believe our approach to the problem has performed well empirically.

Acknowledgement

I would like to thank Prof. Soumen Chakrabarti, Professor, Computer Science and Engineering Department, IIT Bombay for his guidance and valuable inputs given during the course of this project. Prof. Juliana Friere, Professor, Scientific Computing and Imaging Institute, University of Utah is gratefully acknowledged for her insight and assistance throughout the assignment process. Sanjay Agarwal, Principal RSDE, Microsoft Research is thanked for his co-operation in ensuring a smooth integration of our algorithm's results with the Conference Management Toolkit provided by MSR. I am also grateful to Prof. Sundar Vishwanathan, Professor, Computer Science and Engineering Department, IIT Bombay for his assistance in ironing out the difficulties with the Minimum Cost Flow approach. I have had several fruitful discussions with Prof. S. Sudarshan, and Prof. Sunita Sarawagi, professors, Computer Science

and Engineering Department, IIT Bombay regarding alternative formulations and non-linear parameter estimation techniques for the problem. Finally, a very fundamental assumption in the formulation was pointed out to me by Prof. Thorsten Joachims, Professor, Cornell University; 6 outlines relaxations of this assumption. I am also indebted to my colleagues, in particular, Karthik Raman and Nikhil Hooda, senior undergraduates at IIT Bombay for their support.

A PARA : Papers Assigned to Reviewers Apparatus

PARA is a suite of scripts and Java programs, servlets and webpages we developed, designed to handle the assignment task for any conference. Input to the suite is provided in the form of several XML files, detailing reviewer/meta-reviewer information, bids, topics, conflict of interests, paper metadata etc. The output of the suite is an XML file detailing the assignments, the XML schema being compliant with CMT. Several intermediate files are also generated, with the extension ".hsv"(hash separated values) to allow saved changes to be exported from one instance of PARA to another.

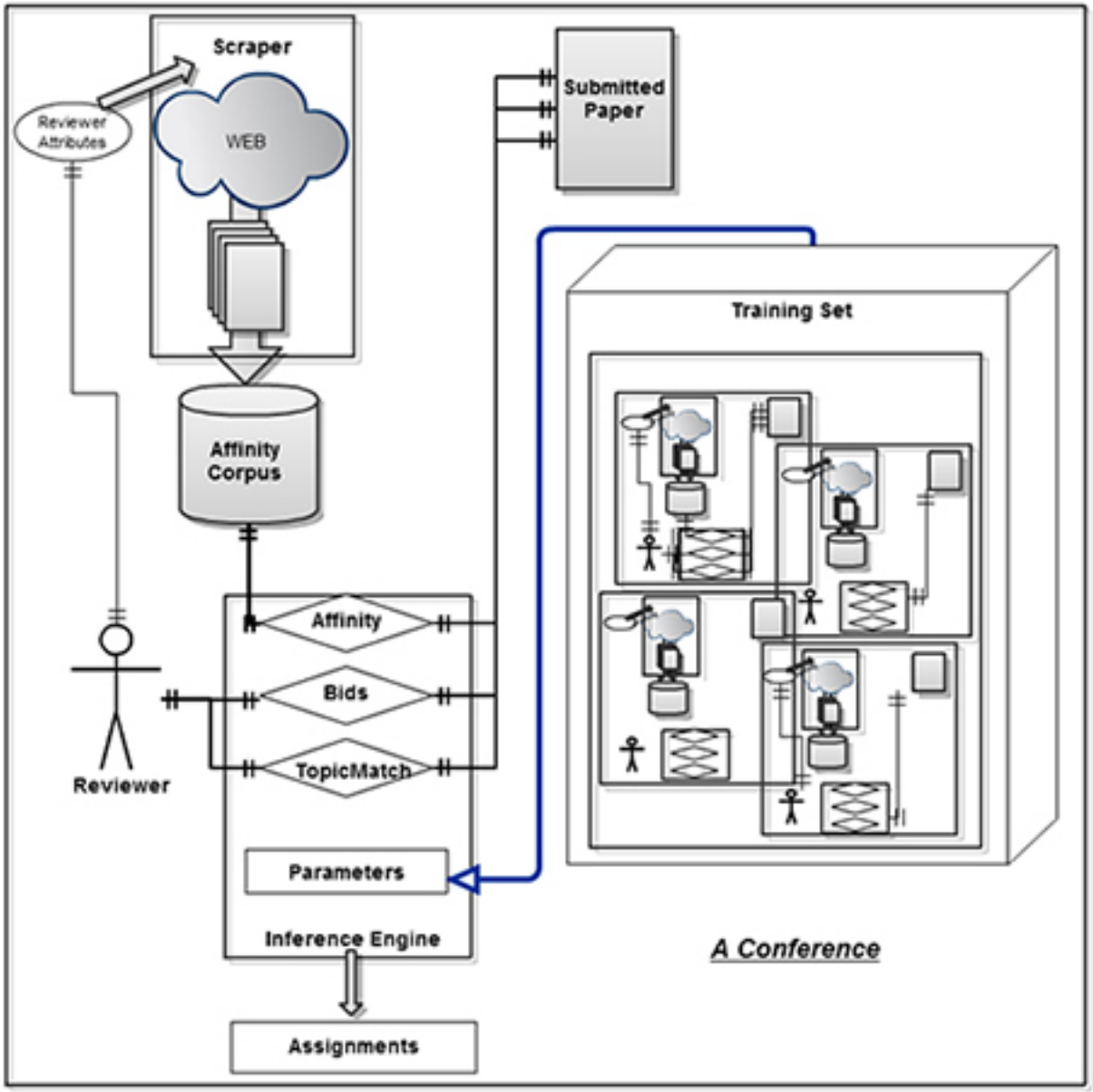

Figure 4: Schematic view of PARA

A.1 Scraper

The input to the Scraper module is a comma-separated-value file (.csv) which details reviewer information for the conference. The output of the Scraper module is a folder hierarchy with the tracks being the uppermost directory, and each reviewer in a particular track having a directory containing *text files* of his recent publications. This is achieved by the following python scripts :

- *JSONScraper.py* : This script queries DBLP with the author name, and then constructs appropriate queries to collect his/her recent papers. Successive requests to Google Scholar are throttled to prevent blacklisting by Google.
- *JSONExceptionScraper.py* : The normal JSONScraper could run into problems if the author's name redirects to a disambiguation page in DBLP. This script handles such "exceptions" and ensures that every reviewer in the conference has an appropriate profile directory. This can also be used to identify dummy reviewers in a conference.
- *Converter.py* : Documents collected from the Web could be in different formats, word documents and PDF files being the most common formats. This script converts documents in various different formats into a standard text format; it uses the *html2text.py* script as an external module. Missing fonts are checked for and prompted on the command line when this script runs.
- *Verifier.py* : This script is not currently used in PARA. It scans through each generated text file, and verifies that the author names (or their variants) match the reviewer name; In essence, it verifies that the tet file indeed belongs to the reviewer profile it has been generated for.
- *SuperDoc.py* : As seen in 3.3, the reviewer profile was assumed to be the concatenation of each collected paper for that reviewer. That task is performed by this script. Several alternate recombinations can be performed by changing this script appropriately.

A.2 SearchEngine

The input to this module is the folder hierarchy output as provided by the Scraper module. The SearchEngine is a combination of a Java application and a Web service, that allows users to index the affinity corpus using various analyzers, and upload documents from remote servers and generate affinity scores between uploaded documents and generated reviewer profiles. It also has the ability to pick up all valid documents in a directory (assuming these valid documents are submitted papers), and generate a ".hsv" file containing the affinity information for these submitted papers and the reviewer profiles indexed in the affinity corpus.

- *TextFileIndexer.java* : This class handles the index generation and analyzer initialization. The included main class allows a command line interface to manipulate the index.

- *QPaS.java* : The query constructor, parser and searcher. This class handles boolean query generation (from the text of the submitted paper), and initializes appropriate result collections to enable searching in the indexed corpus.
- *NovelAnalyzer.java* : We developed a custom analyzer to tokenize the input stream from each document to be indexed in the corpus, and to tokenize each query served to the corpus. This was done to weed out terms in the query with very high document frequency (queries are the full text of submitted papers and can be thousands of terms long).
- *MCFPSolver.java* : This is a Java implementation of the Successive Shortest Paths Algorithm for the Minimum Cost Network Flow. This sets up the network as specified in 3.1 and assigns edge weights with some predetermined parameters $\vec{w}$. It uses the class defined in *MCF.java* to solve the instance.
- *Affinity.java* : Given a directory containing submitted papers, this class constructs queries corresponding to each of the papers and queries the index. Currently, this uses a sequential implementation, but a parallel implementation can be easily achieved. The output of this is a file called "Affinities.hsv" (usually a big text file), listing the affinity scores for each reviewer paper pair (R_i, P_j).

A.3 PARApp

To facilitate user intervention in the assignment process, we developed a graphical user interface tofacilitate dynamic modification of assignments, repeated runs of the MCF solver, easy parameter tuning, load rebalancing, etc. Moreover, most conferences today have a group of overall co-ordinators. It becomes imperative to be able to easily share "partial" assignments. The GUI was coded with these requirements in mind.

- *PARApp.py* : This is the central GUI application. It has a three-pane view, a general view setting up the assignment collaborator's details and the "partial assignment" if any, a per-reviewer view, and a per-paper view. It uses the *notebook.py* module to achieve the desired GUI appearance, and the *MCF.py* to provide a complete python implementation of the Minimum Cost Network Flow framework.
- *XMLCC.py* : This is the control center to handle interchange between XML format and the ".hsv" format used by our application. Several wrappers written to interface our results with Microsoft's CMT will import this module, and this module in turn uses the *ElementTree.py* module to efficiently traverse XML trees.

A.4 Wrappers

Our application, as it stands, cannot replace conference management toolkits popular today. Some severe shortcomings are the lack of group-based access, lack of workflow and unreliable transactions. Hence, we had to build several wrappers to allow our application to interface easily with popular conference management toolkits in use today.

- *BidsParse.py* : This script parses the Bids file as provided by Microsoft's CMT. The input is an XML file with a schema as defined within this script, and the output is an appropriate "Bids.hsv" file. One point to note is that this script does automatic conflict detection, since Microsoft's CMT was found to give a proper subset of our detected conflicts; and we were motivated by a design philosophy of preventing potentially conflicting assignments.
- *ConfidenceCSV.py* : This script mines the assignments performed by CMT (that is, the assignments uploaded by the overall co-ordinators; not the initial assignments as suggested by CMT's internal algorithm) and the associated reviewer confidence. The output is a comma separated value file, and is in a format that can be readily used by the various learning modules.

A.5 coREGLearner

This module is the first approach described in this report. The input to this module is the augmented training dataset, and it outputs the values for the parameters as described in 4.1. This module uses the *JAMA* matrix package, and the *Weka* sourcec code as external libraries. It performs this using the following classes :

- *CoREG.java* : This is the wrapper class that prepares the training data and initializes the two k-NN regressors. Several parameters (for e.g., the 'k' for each regressor, the metrics for each regressor, etc.) can be set in this class. Once the training data is fully labelled, this calls the ordinal logistic regressor.
- *IBkReg.java* : This is the file that actually performs the co-training style algorithm with the two regressors.
- *Zelig* : We use the Zelig package to handle the logistic ordinal regression. This script takes in the fully labelled data and fits the parameters using the logistic ordinal regression model described in 4.1.

A.6 SVMpythonLearner

This module performs the second approach described in this report. The primary file in this module is *ConfPARA.py*, which is used as a custom module to instantiate a version of SVM-struct. This file performs all tasks required for both learning and classification. Appropriate calls to svm_python_learn and svm_python_classify with a simple CSV input file should suffice.

A.7 CitationExtractor

This module generates 0/1 scores that indicate whether a reviewer's paper has been cited by the submitted paper or not. This is achieved using a python script *CiteMatch.py*. This, in turn, uses SeerSuite's ParsCit module, and Stanford NER to perform efficient citation extraction and person name disambiguation.

.

References

[CKR09] Don Conry, Yehuda Koren, and Naren Ramakrishnan, *Recommender systems for the conference paper assignment problem*, RecSys '09: Proceedings of the third ACM conference on Recommender systems (New York, NY, USA), ACM, 2009, pp. 357–360.

[DN92] Susan T. Dumais and Jakob Nielsen, *Automating the assignment of submitted manuscripts to reviewers*, SIGIR, 1992, pp. 233–244.

[FGM05] Jenny Rose Finkel, Trond Grenager, and Christopher Manning, *Incorporating non-local information into information extraction systems by gibbs sampling*, ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (Morristown, NJ, USA), Association for Computational Linguistics, 2005, pp. 363–370.

[GS62] D. Gale and L. S. Shapley, *College admissions and the stability of marriage*, The American Mathematical Monthly **69** (1962), no. 1, 9–15.

[GS07] J. Goldsmith and R. H. Sloan, *The ai conference paper assignment problem*, 2007.

[HJ07] Bert Huang and Tony Jebara, *Loopy belief propagation for bipartite maximum weight b-matching*, Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS), 2007.

[KIL07] Gary King Kosuke Imai and Oliva Lau, *ologit: Ordinal logistic regression for ordered categorical dependent variables*, 2007.

[KV06] Bernhard Korte and Jens Vygen, *Combinatorial optimization: Theory and algorithms*, 3rd ed., Springer, Germany, 2006.

[LJTKJ06] Simon Lacoste-Julien, Ben Taskar, Dan Klein, and Michael I. Jordan, *Word alignment via quadratic assignment*, Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (Morristown, NJ, USA), Association for Computational Linguistics, 2006, pp. 112–119.

[PRtYZ05] Vasin Punyakanok, Dan Roth, Wen tau Yih, and Dav Zimak, *Learning and inference over constrained output*, IJCAI, 2005, pp. 1124–1129.

[SC09] Adith Swaminathan and Soumen Chakrabarti, *Learning many-many matchings*, 2009.

[TCKG05] Benjamin Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin, *Learning structured prediction models: a large margin approach*, ICML, 2005, pp. 896–903.

[THJA04] Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun, *Support vector machine learning for interdependent and structured output spaces*, ICML '04: Proceedings of the twenty-first international conference on Machine learning (New York, NY, USA), ACM, 2004, p. 104.

[ZL07] Zhi-Hua Zhou and Ming Li, *Semisupervised regression with cotraining-style algorithms*, IEEE Trans. on Knowl. and Data Eng. **19** (2007), no. 11, 1479–1493.