

Probabilistic Multi-State Split-Merge Algorithm for Coupling Parameter Estimates

Juan K. Lin

Department of Statistics
Rutgers University
Piscataway, NJ 08854
jklin@stat.rutgers.edu

Abstract

A new approach to finding good local maxima of the likelihood function based on synthesizing information from two local maxima is presented. We investigate the coupled EM algorithm (CoEM) for coupling local maxima solutions from two separate EM runs for the multinomial mixture model. The CoEM algorithm probabilistically splits and merges multiple latent states based on conditional independence assumptions and is numerically shown to significantly improve on uncoupled EM or deterministic annealing (DAEM) parameter estimates.

1 Introduction

The EM algorithm (e.g. Dempster, Laird and Rubin 1977) and its variants are fundamental algorithmic building blocks for parameter estimation in latent variable models. The monotonic convergence property of the EM algorithm, as well its ease of derivation and implementation has made it the principal algorithm for model parameter estimation in the machine learning and data mining communities. The monotonic convergence property guarantees convergence to a local maxima. However, when the likelihood function contains many local maxima, how does one go about fitting the model parameters? A simple strategy is to run the EM algorithm with various initial conditions to map out the local maxima in the likelihood function, and simply choose the best local maxima. This is computationally costly especially in high dimensional data and parameter space. Researchers have tackled the local maxima problem in two ways. One strategy is to modify the likelihood cost function landscape. In analogy with a physical annealing process, the deterministic annealing EM algorithm (DAEM) (Rose et.al. 1990, Ueda et.al. 1998) modifies the likelihood function by adding a temperature control parameter, and explores various annealing schedules to try to find good local maxima. Similarly, the information bottleneck EM algorithm (IB-EM) in-

troduces a general class of cost functions which trade-off information compression and preservation (Elidan and Friedman 2003). A second strategy is to use various criteria for selectively splitting and merging clusters to try to escape from poor local maxima (e.g. Brown et.al. 1992, Ueda et.al. 1998, Jain and Neal 2004). Instead of splitting and merging pairs of states, we present an approach which maps all the latent states of two local maxima parameter estimates to each other. The computational goal is to be able to run multiple EM trials in parallel from different initial conditions, and to synthesize the information from multiple local maxima into better parameter estimates. We investigate the coupled EM algorithm (CoEM) for various multinomial mixture models. Recently, these models have received great interest in the data mining and machine learning communities (e.g. Lee and Seung 1999, Hofmann 2001, Lin 2003, Ding and He 2005).

An outline of the paper is as follows. In Sections 2 and 3, the intuition for combining clusters and the CoEM algorithm for the multinomial mixture model is described. Numerical results are presented in Section 4. The CoEM algorithm for a more complex multinomial mixture traffic model is described in Section 5.

2 Intuition

Our main goal is to find a method of combining information contained in multiple EM *soft* clustering solutions. Some hard classification examples will illustrate the intuition. Given objects in the set $\{NYC, Boston, apple, orange, red, blue\}$, let the first partition solution be $\pi_1 = \{NYC, Boston, apple, orange\}, \{red, blue\}$, and the second partition $\pi_2 = \{NYC, Boston\}, \{apple, orange, red, blue\}$. Even though the two partitions are not optimal, they contain information about the correct underlying classes $\pi_1 \wedge \pi_2 = \{NYC, Boston\}, \{apple, orange\}, \{red, blue\}$. Here $\pi_1 \wedge \pi_2$ denotes the combinatorial meet of the two partitions (e.g. Stanley 1986). Consider a second example of combining

classification information. Let the two partitions be $\pi_1 = \{\text{red, green}\}, \{\text{blue, yellow}\}, \{\text{pacific, atlantic}\}$, and $\pi_2 = \{\text{green, yellow}\}, \{\text{red, blue}\}, \{\text{pacific, atlantic}\}$. The combinatorial join of the two partitions is given by $\pi_1 \vee \pi_2 = \{\text{red, green, blue, yellow}\}, \{\text{pacific, atlantic}\}$, which is the correct underlying classification.

In the first example, taking the meet of the partitions correctly splits up the fruit cluster $\{\text{apple, orange}\}$ from their respective incorrect clusters. In the second example, taking the join of the partitions correctly merges all the colors into one class $\{\text{red, green, blue, yellow}\}$. These examples illustrate some intuition behind how two classification solutions can be combined to form more suitable partitions. Combining information from multiple partitions has been investigated from a lattice theoretic perspective in Neumann and Norton (1986) and Barthelemy et.al. (1986), and from a mutual information perspective in Strehl and Ghosh (2002). In this paper we investigate algorithms for combining two *soft* probabilistic clustering solutions.

3 Coupling EM runs

3.1 Summary of the multinomial mixture model

We wish to formulate an algorithm for combining the information from two EM solutions. In this section, we summarize the multinomial mixture model with one latent variable and a conditional independence assumption. This model has been applied to information retrieval and natural language processing (Brown et.al. 1992, Pereira et.al. 1993, Saul and Pereira 1997, Lee and Seung 1999, Hofmann 2001) and has appeared in the statistics literature as latent class analysis (Everitt 1984). The coupling of EM runs for more structured mixture of multinomials is described in a later section.

Let $\tilde{p}(x, y)$ be the empirically observed joint distribution over discrete random variables X and Y , and let H be a discrete 'class' latent variable. Let the number of states in the random variables be $|X| = |Y| = n$ and $|H| = k$. The model assumes that X and Y are conditionally independent given H , which we write $X \perp Y | H$. Maximizing the likelihood is equivalent to minimizing the following Kullback-Leibler divergence

$$\mathcal{D}(\tilde{p}(x, y) \parallel \sum_h p(x|h)p(y|h)p(h)).$$

with respect to $p(x|h), p(y|h)$ and $p(h)$.

3.2 Initial attempts at coupling EM runs

Suppose two EM algorithms are run in parallel with two different initial conditions. Let H_1 and H_2 be the

latent variables in the two runs. The conditional independence assumptions for the two separate models, $\{X \perp Y | H_1, X \perp Y | H_2\}$, together with the two sets of model parameters specify the marginals $p(x, y, h_1)$ and $p(x, y, h_2)$. We tried embedding the two mixture models of the marginals $p(x, y, h_1)$ and $p(x, y, h_2)$ into a full model of $p(x, y, h_1, h_2)$. First consider the undirected graphical model with edges between X and H_1 , X and H_2 , Y and H_1 , and Y and H_2 . This model graphically links the two underlying graphical models together at the observed variables X and Y . Two undesirable properties of this model are immediately apparent. First this is a loopy graphical model; second, this model's conditional independence assumptions, $X \perp Y | \{H_1, H_2\}$ and $H_1 \perp H_2 | \{X, Y\}$, are not directly consistent with the two multinomial mixture models' conditional independence assumptions. Our second attempt at coupling the mixture models seeks a model of the full joint distribution consistent with the two mixture models. A simple way of modeling $p(x, y, h_1, h_2)$ is to assume $H_1 \perp H_2 | X, Y$, in addition to the two conditional independence assumptions for the two multinomial mixture models $X \perp Y | H_1$, and $X \perp Y | H_2$. This new conditional independence assumption is consistent with the assumptions from the two mixture models since it simply defines the joint as

$$p(x, y, h_1, h_2) = p(h_1|x, y)p(h_2|x, y)\tilde{p}(x, y).$$

Using maximum entropy to pick out a model of the full joint distribution $p(x, y, h_1, h_2)$ given pre-specified marginals $p(x, y, h_1)$ and $p(x, y, h_2)$ selects this exact model. Unfortunately, this is not a graphical model, and parameter estimation is a non-trivial challenge.

3.3 CoEM algorithm for two mixtures of multinomials

Instead of constructing a model for the full joint distribution and worrying about consistency of assumptions, we focus on one-time couplings of parameters from the two mixture models similar to a message passing update. For coupling EM runs, we simply perform a *one-time* "probabilistic split-merge" coupling of two converged EM run parameter estimates, then continue the EM runs separately until convergence. Thus the only difference between regular EM and a coupled EM algorithm is the *one-time* coupling step. This will be referred to as the CoEM algorithm.

We now address the issue of combining parameter estimates. Parameters from the two mixture models $\{p(x|h_1), p(y|h_1), p(h_1)\}$, and $\{p(x|h_2), p(y|h_2), p(h_2)\}$, are assumed to correspond to two local maxima of the likelihood function. To combine information from

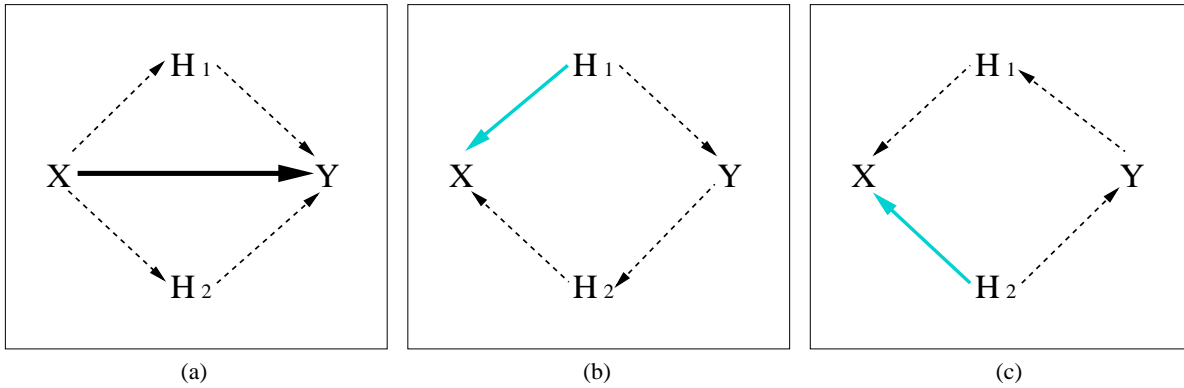


Figure 1: Approximate mapping diagram motivating the coupling updates in the CoEM algorithm. Note, this is not a graphical model diagram.

the two local maxima, we cannot simply take convex combinations of the corresponding parameters. The ordering of the states for H_1 and H_2 are generally not in direct correspondence with each other. Furthermore, some states in H_1 may be mixtures of the states in H_2 and vice versa. In order to compare one set of parameters to the other set, we need a mapping between the states in the two latent variables.

We investigated mappings between latent states for CoEM based purely on conditional independence assumptions. The CoEM algorithm for mixture of multinomials model is as follows:

1 **Iterate** two EM runs in parallel until convergence.

2 **Couple**:

- (a) Assume $H_1 \perp H_2 | Y$,
 compute $p_a(h_1, h_2) = \sum_y p(h_1|y)p(h_2|y)p(y)$.
 Update $p(x|h_1) = \sum_{h_2} p(x|h_2)p_a(h_2|h_1)$,
 and $p(x|h_2) = \sum_{h_1} p(x|h_1)p_a(h_1|h_2)$.

- (b) Assume $H_1 \perp H_2 | X$,
 compute $p_b(h_1, h_2) = \sum_x p(h_1|x)p(h_2|x)p(x)$.
 Update $p(y|h_1) = \sum_{h_2} p(y|h_2)p_b(h_2|h_1)$,
 and $p(y|h_2) = \sum_{h_1} p(y|h_1)p_b(h_1|h_2)$.

3 **Iterate** the two updated EM runs in parallel until convergence.

All conditionals are computed based on the corresponding specified joint distributions. Here the updates for parameters involving X are based on $H_1 \perp H_2 | Y$, while updates for parameters involving Y are based on $H_1 \perp H_2 | X$. This is to retain symmetry in the roles played by X and Y . Figure 1 depicts an approximate

mapping description of the multinomial mixture model, as well as the intuition behind the coupling algorithm. *Note*, these are not graphical model diagrams, as the conditional independence assumptions in CoEM do not exactly correspond to those contained in a DAG interpretation of the diagrams. In Figure 1(a), the dark arrow from X to Y denotes the observed empirical transition mapping $\tilde{p}(y|x)$ obtained from $\tilde{p}(x, y)$. The multinomial mixture model seeks to find compositional mappings from X to H (H_1 and H_2 for the two EM runs) and then Y which approximates $\tilde{p}(y|x)$ optimally in a minimum Kullback-Leibler divergence sense. The updates in CoEM step 2(a) for $p(x|h_1)$ and $p(x|h_2)$ are depicted in Figures 1(b) and (c) respectively.

Other choices of conditional independence assumptions for determining $p(h_1, h_2)$ are explored in sections 4.3 and 4.4. Note that these conditional independence coupling assumptions are used *only* in the coupling step. The **coupling** and **iterate** until convergence steps can be repeated until a satisfactory solution is reached. Except for possible parameters arising from the EM convergence criteria, there are no threshold parameters in the coupling step of the CoEM algorithm to set.

4 Numerical results

We performed numerical experiments comparing regular uncoupled EM runs, coupled EM runs (CoEM), and deterministic annealing EM algorithm runs (DAEM) (Ueda et.al. 1998). In the following subsections, parameter estimation results are reported for synthetically generated data, as well as computer skills, text corpus, and traffic data. Sections 4.2 and 4.3 compare KL-divergence minimization (equivalently likelihood maximization) results. Section 4.4 reports predictive test-set loglikelihood results. We begin with numerical experi-

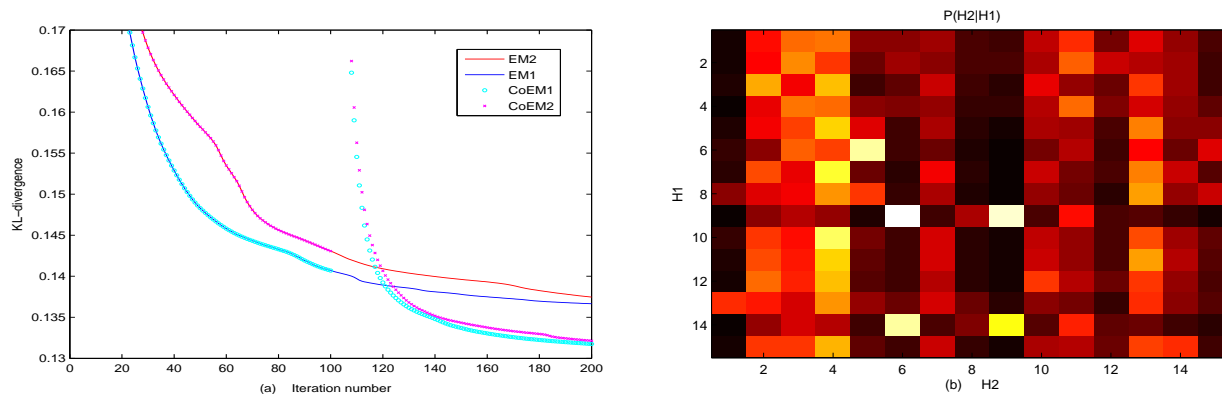


Figure 2: (a) KL-divergence as a function of iteration number for EM and CoEM runs. (b) Transition matrix $p(h_2|h_1)$ used after iteration 100 for CoEM algorithm.

ments on a small computer skills data set to demonstrate the coupling.

4.1 Computer skills data

We applied the CoEM multinomial mixture model algorithm to a computer skills co-occurrence dataset, courtesy of Prof. Richard Martin and the IT consulting firm Comrise. This data set is small and intuitive enough to permit a more detailed analysis of the CoEM algorithm. The raw data consists of a collection of job descriptions, each of which contains a set of computer skills the hiring manager considers important for the job. A co-occurrence matrix is constructed which tabulates the number of times each pair of 159 computer skills appear in a job description. The entries along the diagonal of the co-occurrence matrix contain the number of times each skill occurred over all the job descriptions. The multinomial mixture model was used to find computer skills topic clusters.

Two EM trials are run in parallel with 15 latent states for H for 100 iterations. Subsequently, the two trials are run for an additional 100 iterations both with and without a CoEM coupling step. The CoEM coupling step consists of the updates $p(x|h_2) = \sum_{h_1, y} p(x|h_1)p(h_1|y)p(y|h_2)$, and $p(x|h_1) = \sum_{h_2, y} p(x|h_2)p(h_2|y)p(y|h_1)$. The progression of the KL-divergence is shown in Figure 2(a). The KL-divergence for the EM and CoEM trials are identical for the first 100 iterations. For CoEM, the coupling step after iteration 100 increases the KL-divergence, though after a few additional iterations, the CoEM trials arrive at significantly lower KL-divergences compared with their uncoupled EM counterparts.

The coupling transition matrix $p(h_2|h_1) = \sum_x p(h_2|y)p(y|h_1)$ used after iteration 100 is depicted

in Figure 2(b). We focus on state $H_1 = 1$. From CoEM step 2(a), the updated $p(x|h_1 = 1)$ is a convex combination of the distributions $p(x|h_2)$ with weights given by the first row of the transition matrix in Figure 2(b). The coefficients in $p(h_2|h_1 = 1)$ has large contributions for $h_2 = \{3, 4, 11\}$. Though this is a soft probabilistic clustering model, we look at the MAP assigned states to help us understand how the CoEM algorithm couples the EM runs. MAP assignment for latent state $H_1 = 1$ (state 1 in EM run 1) gives the computer skills $\{pc(ibm), windows95, msoffice, dos\}$ after both 100 and 200 iterations for EM run 1. After 100 iterations, latent state $H_2 = 3$ contains skills $\{windowsnt, windows95, visualc++, visualbasic\}$, $H_2 = 4$ contains $\{sunos, solaris, unix\}$, and $H_2 = 11$ contains $\{pc(ibm), msoffice, msoffice97, msproject\}$. After 200 iterations, the CoEM run for H_1 gives MAP assigned skills for $H_1 = 1$ of $\{pc(ibm), windows95, windowsnt, dos, visualc++, visualbasic, vbscript\}$, while the uncoupled EM run 1 retains the same MAP assigned skills $\{pc(ibm), windows95, msoffice, dos\}$. The coupling step merged the skills $\{windowsnt, visualc++, visualbasic, vbscript\}$ into cluster $H_1 = 1$, and split off the skill $\{msoffice\}$ into a new cluster containing $\{msoffice, msoffice97, msproject, msexchange\}$. The coupling of other latent states showed similar behavior. Intuitively, large coefficients across a row in Figure 2(b) give rise to a merging of states, while large coefficients across a column lead to splitting of states. Thus, in contrast to pairwise merge, or single split algorithms, CoEM can merge more than two states, and split a single state into multiple states.

4.2 Synthetic data

We synthetically generated both exactly decomposable and noisy empirical joint distributions $\tilde{p}(x, y)$.

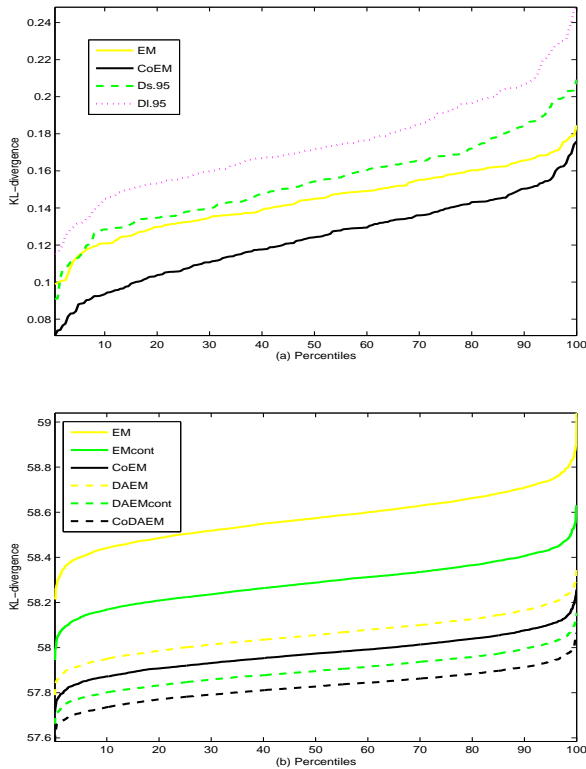


Figure 3: KL-divergence percentile plots for (a) exactly decomposable $\tilde{p}(x, y)$ (Table 1), (b) randomly generated $\tilde{p}(x, y)$. The percentiles for $Ds.9$ and $Dl.9$ are significantly larger, and do not appear on the plots.

For exactly decomposable $\tilde{p}(x, y)$, we randomly sampled $p(x|h)$, $p(y|h)$ and $p(h)$ to construct $\tilde{p}(x, y)$ from the multinomial mixture model. Even though the minimum KL-divergence is zero for these distributions, it is *frustratingly difficult* for the EM algorithm to find the exact decomposition. The noisy distributions were randomly sampled from the $n^2 - 1$ dimensional simplex.

The experimental setup is as follows. Multiple pairs of EM runs were run for a suitably large number of EM iterations. Instead of using convergence conditions, we simply chose a fixed number I of iterations for each EM run. This was motivated by the observation of many plateaus in the likelihood landscape. Also, this fixes the number of iterations in the comparisons between the various algorithms. We assume that the iterations have converged to local maxima solutions at that time. For the CoEM runs, we coupled pairs of EM runs, then iterated them for an addition I iterations. For EM runs (“EM continued”), we did not perform the coupling step, and simply continued for an additional I iterations.

Testing of the DAEM algorithm required the specification of annealing schedules. We implemented a linear annealing schedule where the inverse temperature parameter β was linearly ramped from β_{min} at the first iteration to 1 at the $2I$ -th iteration. We also tried a stepwise constant annealing schedule with $\beta = \beta_{min}$ for the first $2I/3$ iterations, $\beta = (1 + \beta_{min})/2$ for the next $2I/3$, and regular EM iterations ($\beta = 1$) for the remaining $2I/3$ iterations. The EM, CoEM, and DAEM runs all consisted of a total of $2I$ iterations.

Table 1: KL-divergence percentiles for $n = 200, k = 25$

	EM	CoEM	Dl.9	Dl.95	Ds.9	Ds.95
1%	.099	.074	3.11	.116	3.19	.091
5%	.114	.088	3.13	.132	3.22	.113
10%	.121	.093	3.14	.145	3.22	.128
time	9.47	9.47	17.20	17.40	14.62	14.60

We ran 200 pairs of EM runs (400 runs) initialized with random initial values with the iteration count $I = 300$. Cumulative percentiles of final KL-divergences for exactly decomposable empirical joint distributions with $n = 200$ and $k = 25$ are tabulated in Table 1 and plotted in Figure 3(a). The DAEM algorithm with both linear and stepwise constant annealing schedules were tested for various settings of β_{min} . The columns $Dl.9$, $Dl.95$, $Ds.9$ and $Ds.95$ correspond to linear (Dl) and stepwise constant (Ds) annealing schedules at values of $\beta_{min} = .9$ and $.95$. DAEM runs with $\beta_{min} < .9$ resulted in very poor parameter estimates. The DAEM trials were initialized with the same initial conditions as the EM and CoEM trials. From Table 1 we see that the percentiles are the smallest for the CoEM algorithm.

Computation time in seconds needed to run the corresponding algorithms for $2I = 600$ iterations on a Pentium-IV 1.7 GHz computer are also listed in Table 1. Since the CoEM runs consist of a single simple coupling step after iteration 300, and regular EM iterations otherwise, the computation time for CoEM is essentially equivalent to regular EM algorithm. On the other hand, DAEM iterations require componentwise exponentiation of the parameters and result in significant increases in computation time. The Ds DAEM runs are less costly computationally than the Dl runs because the last 200 iterations of the Ds runs are regular EM iterations. From the numerics, the performance of the DAEM runs depended critically on the annealing schedules. The search for an annealing schedule which can successfully maneuver around poor local maxima in the loglikelihood is a considerable challenge. All the DAEM runs except $Ds.95$ performed worse than plain

EM runs, and in addition, computation cost is significantly higher for DAEM than for CoEM.

We then tested the performance of the algorithms for randomly generated noisy empirical joint distributions $\tilde{p}(x, y)$. In Figure 3(b), the KL-divergence percentiles are plotted for 200 pairs of runs for a few different algorithms. We increased the base number of EM iterations to $I = 600$. For reference, lines labeled EM and DAEM are KL-divergence percentiles after I iterations, before the coupling step. The line EMcont is for $2I$ uncoupled iterations of EM. For the DAEMcont algorithm, regular DAEM iterations are run for the first I iterations, and continued with regular EM for another I iterations. Using the annealing analogy, the DAEMcont runs were not “re-heated”. Experiments with “re-heated” DAEM runs performed worse than DAEMcont. Comparing the percentiles for EMcont, CoEM and DAEM, we see improvements of CoEM over EMcont, and DAEM ($Ds.95$) over CoEM. The difference in performance of CoEM and DAEM seems to depend on the actual structure (decomposable vs. noisy) of the empirical joint distribution. Since the CoEM algorithm simply couples two local maxima parameters, we also implemented a CoDAEM algorithm which couples the local maxima found using the DAEM ($Ds.95$) algorithm. Essentially, the coupling step in CoEM can be implemented to couple local maxima found using any optimization algorithm. From the figure, the CoDAEM algorithm gives better percentile performance than the uncoupled DAEMcont runs.

4.3 Text corpus data

We compared the results of running EM, CoEM and DAEM for multinomial mixture models on text corpus data. Non-negative matrix factorization (NMF), or equivalently pLSA have been applied to decompose document-word co-occurrence matrix. In this context the latent class model pLSA does not provide a probabilistic generative model, and does not cleanly assign predictive probabilities to new documents. A generative model called Latent Dirichlet Allocation is described in Blei et.al. (2003), along with a discussion of some issues with computing perplexity scores with pLSA. Here we compare results from a pure likelihood maximization or equivalently KL-divergence minimization perspective, essentially seeking the best NMF factorization of the document-word co-occurrence matrix using the KL-divergence measure of distance. We compared the various algorithms on the Medlars document collection with 1033 medical abstracts, and the Cranfield collection of 1398 aerodynamics abstracts.

We ran the 25 pairs of EM runs using random ini-

tializations for $I = 50$ iterations each (labeled *EM*). For the CoEM runs, we mapped pairs of parameter estimates to each other using various conditional independence assumptions as follows. For *CoEM_a*, we assumed the the latent variables for the two runs are conditionally independent given the W , or in compact graphical model notation $H_1 - W - H_2$. The *CoEM_b*, *CoEM_c* and *CoEM_d* couplings assume respectively, $H_1 - D - H_2$, $H_1 - W - D - H_2$, and $H_1 - D - W - H_2$, where D is the document and W the word discrete variables. For each of the CoEM runs, the respective conditional independence assumptions are used to specify the conditionals $p(h_2|h_1)$ and $p(h_1|h_2)$ from the two sets of parameters. These conditionals are used in the coupling step for the respective CoEM runs. After the coupling step, plain EM is run for an additional $I = 50$ iterations for each run. The original EM runs are also continued for a total of 100 iterations without coupling (labeled *EMcon*). For comparison, the best performing DAEM algorithm with $\beta = .95$ for the first 33 iterations, $\beta = .975$ for the next 33, and plain EM for the remaining 34 iterations is also run using the same initial conditions. The final KL-divergences for all 25 pairs of runs (50 total) for each algorithm are ranked in order of KL-divergence scores and plotted in Figures 4(Medline) and 5(Cranfield) for $k = 25, 50$ and 100. In the figures, dotted lines represent *EM* runs, the dashed line for *DAEM*, and solid lines for the *CoEM* runs. The training set likelihoods are inverse monotonically related to these KL-divergences. From the figures, as expected, 100 iteration *EMcon* runs improve over 50 iteration *EM* runs. The stepwise constant annealing schedule *DAEM* runs perform better than *EMcon*. However, all the coupled EM runs *CoEM_a* through *CoEM_d* perform significantly better than both *EMcon* and *DAEM*. For $k = 50$ and $k = 100$, every *CoEM_{cd}* run achieves better KL-divergence (equivalently log-likelihood) scores than the best *EMcon* and *DAEM* runs. It is interesting to note that *CoEM_a* performs better than *CoEM_b*, and the stronger coupling assumptions in *CoEM_c* and *CoEM_d* both perform better than *CoEM_a* and *CoEM_b*.

We performed a subsequent experiment to determine whether a split-and-merge “self-coupling” version of CoEM can account to the performance gains in the CoEM runs. In Figure 6, the coupling steps in the *CoEM_b* and *CoEM_d* runs were all based on one set of multinomial mixture parameters, and re-labeled *SCoEM_b* and *SCoEM_d*. The roles played by the two sets of parameters in those runs were all played by a single set. The splitting and merging are all based on the latent states for one latent variable, with similarity of the latent states based on the coupling assumption. This “self-coupled” EM algorithm is analogous to the

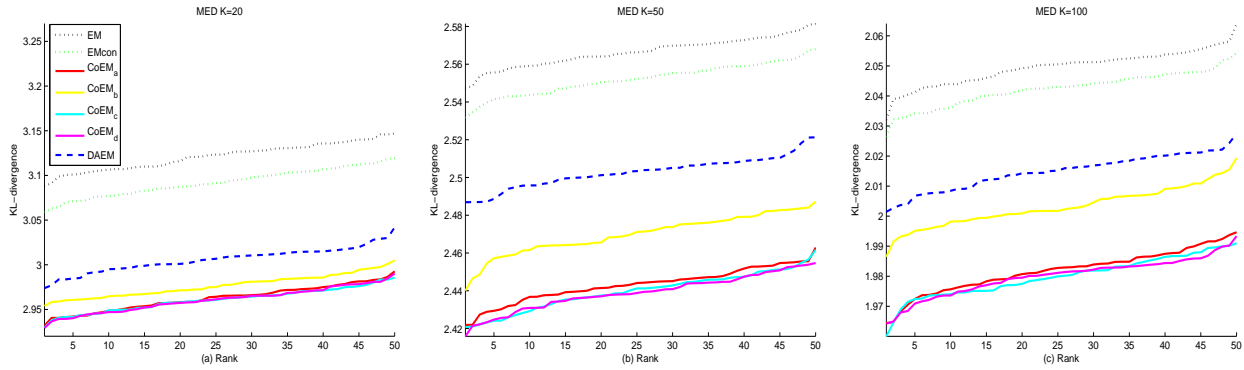


Figure 4: Final KL-divergences for the Medline document collection.

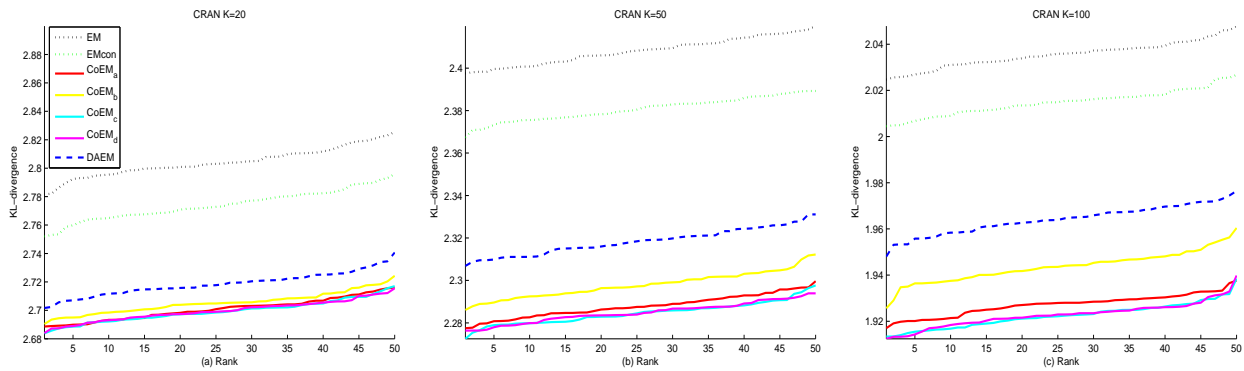


Figure 5: Final KL-divergences for the Cranfield document collection.

split-and-merge algorithm which is based on a single set of parameter estimates. As seen in the figure, the self-coupled *SCoEM* runs all achieve better parameter estimates than *EMcon*. Nevertheless, the *CoEM_a* and *CoEM_c* parameter estimates achieved by combining *two* sets of runs give the two best (lowest) KL-divergences percentile curves, pointing to an empirical success in synthesizing information from two local maxima of the likelihood.

4.4 Traffic between Autonomous Systems

To compare predictive test-set log-likelihoods of the various algorithms we switched to a setting where the mixture of multinomials latent class model provides a probabilistic model of the data. We consider the mixture of multinomials as providing a model of source destination traffic. Here $\tilde{p}(x, y)$ is the empirical traffic distribution from source x to destination y . The parameters $p(h)$ contain the distribution of traffic through latent “hubs”, while $p(x|h)$ describes the onramp traffic distribution from source x to hub h , and $p(y|h)$ the

offramp traffic to destination y from hub h . In this setting the the sources/destinations constitute a fixed set, and the traffic models define generative probabilities for new traffic between the sources and destinations. The motivation for the CoEM algorithm is intuitive in this traffic model setting. The defined $p_a(h_1, h_2)$ describes the traffic between hubs in H_1 and H_2 based on taking offramps from H_1 to destination Y , then onramps from Y to H_2 .

We analyzed internet topology data as reflected in a connectivity graph between Autonomous Systems (AS). The data consists of AS paths in BGP routing tables collected by the server *route-views.oregonix.net*. This data is the basis of the power-law analysis in Qien et.al. (2002) and is publicly available at topology.eecs.umich.edu/data.html. The AS connectivity graph is a symmetric binary matrix. After trimming out nodes with no connections, we are left with an undirected binary AS connectivity graph with 13233 interconnected AS nodes and 55448 edges.

We simulated random walk traffic on this AS connectivity graph. For the training set, we performed

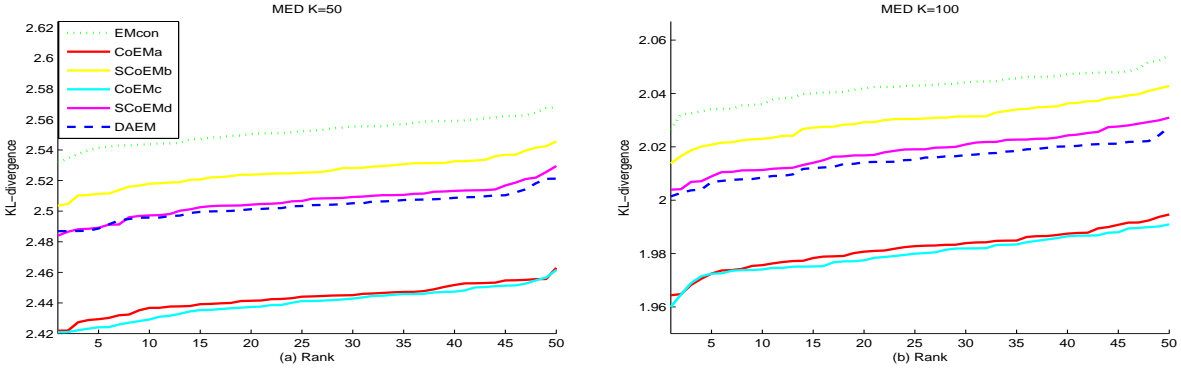


Figure 6: Final KL-divergences for the Cranfield document collection.

100000 *single* random walk steps on the connectivity graph. The 100000 source nodes were sampled in accordance with the stationary distribution of the random walk on the connectivity graph. Traffic from source nodes are assumed to follow each of the outgoing edges with equal probability. Since multinomial mixture models can be prone to overfitting problems, we added a single pseudo-count traffic for each edge in the connectivity graph. If traffic from a source to a destination is not observed in the test set, but appears in the training set, the traffic models may assign zero probability to the test set likelihood. Early stopping effectively stops parameter updates if an update assigns zero probability to a traffic path that appears in the test set. In Hofmann (2001) an additional inverse annealing (heating) is used to smooth multinomial parameters and prevent sparseness. For the test set, 20000 single random walk steps were sampled.

Results using the same algorithmic settings as the previous section are shown in Figure 7, with dotted lines for *EM* runs, solid lines for the *CoEM* runs, and the dashed line for *DAEM* runs. The results are very similar to the KL-divergence results for the Medline and Cranfield document collections. The *CoEM* runs, in particular, *CoEM_c* and *CoEM_d*, consistently achieve the best test-set log-likelihoods. *CoEM* does not seem to have any worse over-fitting problems than regular *EM*. The probabilistic mapping coupling step may in fact help smooth the multinomial distributions.

5 Structured Multinomial Model

We wished to explore the coupling of *EM* runs for other multinomial models. We considered a more complex multinomial model of traffic (Lin 2006). The starting point of the analysis is traffic data consisting of n_{ij} counts of traffic from source $X = i$ to destination $X' = j$. We assume that all sources are destinations, and

destinations sources. Discrete latent variables H and H' are introduced which characterize the underlying entrance hubs and exit hubs on the highway. We assume that all entrances are exits, and vice versa. Our model of traffic flow consists of onramp traffic from sources to highway entrances, highway traffic from entrances to exits, and offramp traffic from highway exits to destinations. The model assigns a probability of going from source i to destination j of:

$$p(i, j) = \sum_{k, l} \alpha_{ik} \beta_{kl} \gamma_{jl},$$

where $\alpha_{ik} = P(X = i | H = k)$, $\beta_{kl} = P(H = k, H' = l)$, and $\gamma_{jl} = P(X' = j | H' = l)$. In words, α_{ik} is the fraction of traffic at entrance k from source i , β_{kl} is the probability of going from entrance k to exit l on the highway, and γ_{jl} is the fraction of traffic at exit l that proceed to destination j . The double sum in the expression is over all highway entrances and exits. Note that the traffic model is probabilistic, and in general allows for more than one highway route from source to destination. We further impose a constraint equating the onramp and offramp traffic distributions: $\gamma_{jl} = \alpha_{jl}$. Thus the fraction of traffic at exit l which continue to destination j is equal to the fraction of traffic at entrance l which originate from j . The model parameters are specified by $\alpha(x|h) = P(x|h)$ and $\beta(h, h') = P(h, h')$, which specify respectively the onramp/offramp traffic distribution, and highway traffic between the entrances and exits. Let the total amount of observed traffic be $N = \sum_{i, j} n_{ij}$, and let $\tilde{p}_{ij} = n_{ij}/N$ be the observed empirical joint distribution $\tilde{p}(x = i, x' = j)$.

The log-likelihood function is given by

$$\mathcal{L} = N \sum_{x, x'} \tilde{p}(x, x') \log \left[\sum_{h, h'} \alpha(x|h) \beta(h, h') \alpha(x'|h') \right].$$

Maximizing the likelihood of the observed source-destination traffic counts is equivalent to minimizing the

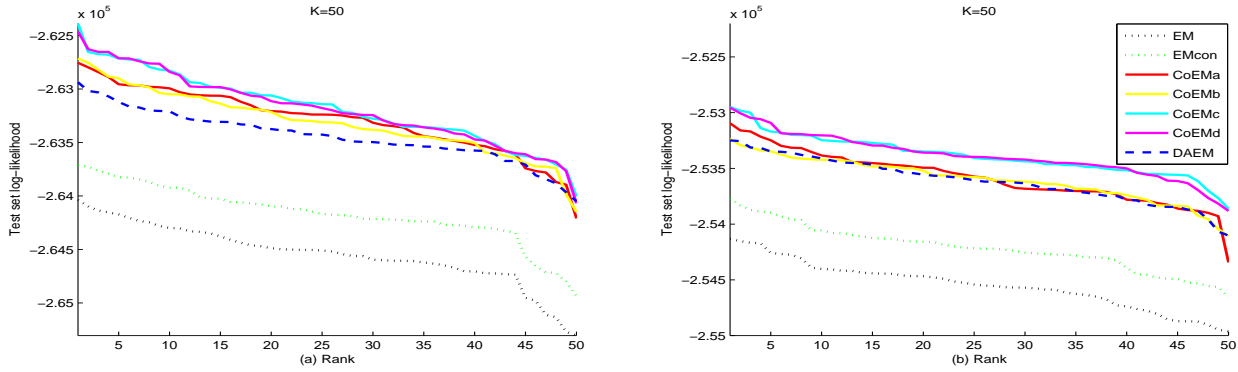


Figure 7: Test set log-likelihoods for the ASN traffic data set.

following Kullback-Leibler divergence:

$$\mathcal{D}(\tilde{p}(x, x') \parallel \sum_{h, h'} \alpha(x|h)\beta(h, h')\alpha(x'|h')).$$

The EM algorithm gives the following update equations
E-step

$$q(h, h'|x, x') = \frac{p(x, x', h, h')}{\sum_{h, h'} p(x, x', h, h')}$$

where $p(x, x', h, h') = \alpha(x|h)\beta(h, h')\alpha(x'|h')$.

M-step

$$\alpha(x|h) = \frac{\tilde{p}(X = x, H = h) + \tilde{p}(X' = x, H' = h)}{\tilde{p}(H = h) + \tilde{p}(H' = h)},$$

$$\beta(h, h') = \tilde{p}_{hh'},$$

where \tilde{p}_{xh} , $\tilde{p}_{x'h'}$, \tilde{p}_h , $\tilde{p}_{h'}$, and $\tilde{p}_{hh'}$ are the corresponding marginals of $\tilde{p}_{xx'}q(h, h'|x, x')$.

A coupled version of the EM algorithm based on the mapping diagram in Figure 8(a) is implemented and compared to uncoupled EM runs in Figure 8(b)(c). Parameters are updated as follows: $p(h_1|h_2) = \sum_x \alpha(h_1|x)\alpha(x|h_2)$, $\alpha(x|h_2) = \sum_{h_1} \alpha(x|h_1)p(h_1|h_2)$, and $\beta(h_1, h'_1) = \sum_{h_2, h'_2} p(h_1|h_2)\beta(h_2, h'_2)p(h'_1|h'_2)$. A total of 25 pairs of EM runs were run for 100 iterations with (CoEM) and without (EMcon) coupling after 50 iterations. Consistent with the previous numerical experiments, the CoEM runs significantly outperform uncoupled EM runs.

6 Discussion

Various multinomial mixture models and their associated matrix factorization algorithms have generated great interest in the machine learning and data mining communities. However, the problem of finding good parameter estimates (factorizations) cannot be overlooked. This paper presents a simple heuristic algorithm

(CoEM) for coupling two local maxima solutions. The coupling step is similar in spirit to split-and-merge algorithms (Brown et.al. 1992, Ueda et.al. 1999, Jain and Neal 2004). An important distinction is the probabilistic mapping based on conditional independence assumptions used in the coupling step for CoEM. This results in a probabilistic split-and-merge coupling between *all* latent states in the two local maxima parameter estimates. Thus, there can be a merging of more than two states, and a splitting into more than two states. In contrast, the split-and-merge algorithms are based on *pairwise* merges and two-way splits. In addition, the CoEM coupling does not have any splitting or merging threshold parameter. Comparing the CoEM algorithm to the computationally more expensive DAEM (Ueda et.al. 1998) algorithm, the CoEM algorithm attempts to synthesize information from multiple local maxima, whereas the DAEM algorithm tries to bypass poor local maxima by smoothing out the likelihood landscape. Performance of the DAEM algorithm depends critically on the annealing schedule, whereas the CoEM algorithm has no annealing schedule to explore or control parameters to set. In the numerical experiments presented in this paper, the CoEM algorithm obtained the best parameter estimates. Very significantly, in our experiments on text corpus data, *every* CoEM_{cd} run resulted in better parameters than the *all* EM and DAEM runs.

As presented in this paper, the CoEM algorithm is a heuristic algorithm like the various split and merge algorithms. The focus of this paper is on numerical justifications of the CoEM algorithm when contrasted with uncoupled EM and DAEM algorithms. Comparisons of final parameter estimates in terms of loglikelihood (equivalent KL-divergence) and predictive test set loglikelihood scores show significant improvements of CoEM over EM and DAEM. Comparisons in terms of computation time is favorable for the CoEM algorithm

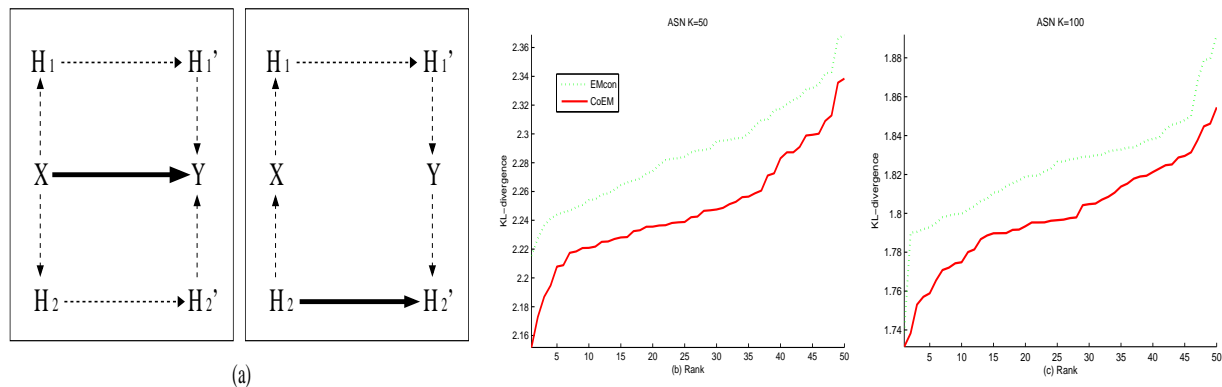


Figure 8: Mapping diagram and final KL-divergences for the ASN traffic data set.

since it is just regular EM with a low cost coupling step. Simplicity of the CoEM algorithm also means there is no exploration needed, whereas DAEM requires fine tuning of annealing schedules, and SMEM requires the setting of splitting and merging criteria. Recurring results such as improvements of CoEM over EM, and better performances of $CoEM_c$ and $CoEM_d$ parameter estimates over $CoEM_a$ and $CoEM_b$ hint at an underlying geometric relationship between the local maxima. We are pursuing information theoretic motivations and derivations of CoEM algorithms for more general classes of models such as mixtures of Gaussians and graphical models with multiple latent variables. The algorithm can also be extended to *repeated* couplings of pairs of EM runs, and couplings of more than two EM runs at a time.

7 Acknowledgments

This work was supported by the National Science Foundation (NSF Grant DMS-0312275).

References

Barthlemy, J.P, Leclerc, B., Monjardet, B. (1986), On the use of ordered sets in problems of comparison and consensus of classifications, *Journal of classification*, 3, 187-224.

Blei, D., Ng, A. and Jordan, M. (2003) Latent Dirichlet Allocation, *The Journal of Machine Learning Research*, vol 3, pp.993-1022

Brown, P., Della Pietra, V., deSouza, P., Lai, J. (1992). Class-based n-gram models of natural language. *Computational Linguistic*, 18:467-479, 1992.

Dempster, A., Laird, N. and Rubin, D. (1977) Maximum likelihood from incomplete data via the EM

algorithm. *J. Roy. Stat. Soc. B* 39:1-39, 1977.

Ding, C. and He, X. (2005) On the Equivalence of Non-negative Matrix Factorization and Spectral Clustering. *Proc. SIAM International Conference on Data Mining (SDM'05)*.

Elidan, G. and Friedman, N. (2003) The Information Bottleneck EM Algorithm. In *UAI 2003*.

Everitt, B. (1984). *An Introduction to Latent Variable Models*. London: Chapman & Hall.

Hofmann, T. (2001) Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42, 177-196.

Jain, S. and Neal, R. M. (2004) A Split-Merge Markov Chain Monte Carlo Procedure for the Dirichlet Process Mixture Model, *Journal of Computational and Graphical Statistics*, vol. 13, pp. 158-182.

Lee, D. and Seung, S. (1999) Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(675), 788-791.

Lin, J. (2003) Reduced Rank Approximations of Transition Matrices, in C. M. Bishop and B. J. Frey (eds), *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, Jan 3-6, 2003, Key West, FL.

Lin, J. (2006) Traffic Models for Community-based Ranking and Navigation", to appear, *Lecture Notes in Computer Science: Workshop on Internet and Network Economics*, Springer-Verlag.

Neumann, D. A., Norton, V. T. (1986). On lattice consensus methods. *Journal of Classification*, 3:225-256.

Pereira, F., Tishby, N., and Lee, L. (1993) Distributional clustering of English words. In *Proceedings of the ACL*, pp183-190, 1993.

Rose, K., Gurewitz, E., and Fox, G. (1990) A deterministic annealing approach to clustering. *Pattern Recognition Letters*, 11(11):589-594, 1990.

Saul, L. and Pereira, F. (1997) Aggregate and mixed-order Markov models for statistical language processing. In *Proceedings of the second conference on empirical methods in natural language processing*, 81-89.

Stanley, R. (1986) *Enumerative Combinatorics*, Wadsworth and Brooks/Cole, Monterey, CA, 1986.

Strehl, A. and Ghosh, J. (2002) Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal on Machine Learning Research (JMLR)*, 3:583-617, December 2002.

Ueda, N. and Nakano, R. (1998) Deterministic annealing EM algorithm *Neural Networks* Vol 11, Issue 2, pp271-282, March 1998.

Ueda, N. and Nakano, R., Ghahramani, Z. and Hinton, G. (1999). SMEM algorithm for mixture models. In *Advances in Neural Information Processing Systems 11, NIPS 98*, 1999.

Qien, C., Chang, H., Govindan, R., Jamin, S., Shenker, S. and Willinger, W. (1992) The Origin of Power Laws in Internet Topologies Revisited, *Proc. of IEEE Infocom, 2002*.