The Gardener's Problem for Web Information Monitoring

Byron J. Gao[1,2], Mingji Xia[2], Walter Cai[3], David C. Anastasiu[1]
[1] Department of Computer Science, Texas State University – San Marcos, San Marcos, TX, USA
[2] Computer Science Department, University of Wisconsin – Madison, Madison, WI, USA
[3] Memorial High School, Madison, WI, USA
bgao@txstate.edu, xmjljx@gmail.com, walter.cai1@gmail.com, da1143@txstate.edu

ABSTRACT

We introduce and theoretically study the Gardener's problem that well models many web information monitoring scenarios, where numerous dynamically changing web sources are monitored and local information needs to be periodically updated under communication and computation capacity constraints. Typical such examples include maintenance of inverted indexes for search engines and maintenance of extracted structures for unstructured data management systems. We formulate a corresponding multicriteria optimization problem and propose heuristic solutions.

Categories and Subject Descriptors: H.1.0 [Information Systems]: Models and Principles – *General*

General Terms: Theory, Algorithms, Performance

1. INTRODUCTION

A gardener has n plants to maintain. Each plant labeled i needs to be maintained at least once every w_i days and each maintenance costs c_i workload. The gardener has a daily workload capacity of k on any day and $\overline{k} \leq k$ on average. Assuming all plants were maintained yesterday, find a feasible schedule starting from today until forever so that all plants are always maintained in time.

The Gardener's problem well models many information monitoring scenarios. Today we have unprecedented access to information. Information depreciates in value with time. Given a large number of dynamically changing sources to monitor, it is important and challenging to make decisions that keep the information as current as possible yet using as few as possible resources [3].

The World Wide Web consists of a huge collection of decentralized pages that are modified at random times. Thus, web information monitoring applications abound. For example, maintenance of inverted indexes for search engines. Due to the explosive growth of the web, crawling web pages has become increasingly challenging. It would take up to 6 months for a new page to be indexed by popular search engines [4]. To substantially improve up-to-dateness of inverted indexes and save on network bandwidth, incremental crawling [4, 8] was introduced, where crawlers estimate how often pages change, and then *schedule* pages for revisit based on their estimated change frequency and importance.

As another example, extracted structures (e.g., ER graphs) in unstructured data management systems (e.g., DBLife [7] developed in the context of community information management) need to be maintained periodically. The web is a colossal free text publishing platform. Today the majority of data appear unstructured and managing unstructured data represents the largest opportunity since managing relational data. An essential step in unstructured data management is to extract structures embedded in unstructured data, enabling structured queries. Web sources are highly dynamic, maintaining extracted structures is a labor intensive undertaking, and developing *scheduling* techniques to improve up-to-dateness and reduce maintenance cost is critical [7].

From the above-defined Gardener's decision problem, various optimization versions can be derived based on tardiness and workload. The measure *tardiness* is either 0 (maintained in time) or a positive number indicating the number of days after the due day of a maintenance. Parameter w_i is the size of the *maintenance window* for plant i. The notion of *day* here is symbolic, it can be time slot of any length. A *feasible* schedule is one satisfying all the given constraints.

Let $K(d)$ be the workload of the gardener at day d, then the daily average workload $\overline{K} = \lim_{t\to\infty} \frac{1}{t} \sum_{d=1}^{t} K(d)$. The inclusion of $\overline{k}$ in the problem definition allows us to derive an optimization criterion that minimizes the total workload, which is equivalent to minimizing $\overline{K}$.

For the optimization version, we formulate a multicriteria problem minimizing tardiness and workload simultaneously. The Gardener's problem is about maintenance scheduling. While existing maintenance scheduling problems are single objective [1, 2], real applications often involve several incommensurable objectives that need to be optimized simultaneously. Taking into account of several criteria enables us to propose more realistic solutions to the decision maker [11].

Multicriteria problems do not have a single optimal solution. Instead, all non-dominated solutions form an optimal Pareto set. The entire problem solving procedure usually involves a human decision maker, who evaluates the many alternatives in the Pareto set and chooses the best trade-off. Thus, beyond providing heuristics that generate alternative schedules, we further utilize OLAP technology, allowing decision makers to interactively evaluate these schedules by navigating in a simulation cube.

CIKM'09, November 2–6, 2009, Hong Kong, China.

2. RELATED WORK

Web Information Monitoring. Although not a technical paper, [3] discusses the opportunities and challenges in web information monitoring in a convincing manner.

[4, 8] propose incremental crawlers in an effort to keep inverted indexes fresh. They do not discuss scheduling. [5] theoretically studies a crawler scheduling problem minimizing the fraction of time pages spend out of date, assuming Poisson page change processes and a general distribution for page access time. We do not make such assumptions and we focus on multicriteria maintenance scheduling.

Unstructured data management systems, e.g., DBLife [7], provide an excellent testbed for studying web information monitoring and maintenance scheduling.

Maintenance Scheduling. Scheduling theory first appeared in the mid 1950's. Scheduling concerns the allocation of limited resources to tasks over time and is normally formulated as optimization problems [10, 6].

Maintenance scheduling is a non-typical category of scheduling problems. Instead of completion time, it concerns periodic in-time maintenance of jobs over a long time span. [1, 2] theoretically studied the windows scheduling problem, which is similar to the Gardener's problem but without the specification of $\overline{k}$. By including $\overline{k}$, the Gardener's problem allows as to derive optimization problems minimizing total workload, which is much desirable in web information monitoring, e.g., to save on total bandwidth. Their problems, however, minimize the maximum daily workload over all days, which does not directly lead to minimized average daily workload and equivalently total workload.

Multicriteria Maintenance Scheduling. Multicriteria optimization has been studied for decades [9]. An excellent overview for multicriteria scheduling can be found in [11]. Maintenance scheduling has not been extensively studied. To our knowledge, multicriteria maintenance scheduling has not been explored previously.

3. THEORETICAL RESULTS

In this section, we study interesting properties of the Gardener's problem, its NP-hardness and approximation.

Notations. We use a tuple $(k, \overline{k}, (c_1, w_1), \ldots, (c_n, w_n))$ to denote an instance of the Gardener's problem. In the instance, each plant i has a *maintenance window* of size w_i and maintenance cost of c_i. The *maintenance window constraint* requires a plant to be maintained at least once within its arbitrarily located maintenance window.

We use $K(d)$ to denote the workload of the gardener at day d, K_{max} to denote the largest daily workload over all days. Then the total workload of the gardener in the first t days $\mathbb{K}(t) = \sum_{d=1}^{t} K(d)$, and the average daily workload $\overline{K} = \lim_{t\to\infty} \frac{1}{t}\sum_{d=1}^{t} K(d) = \lim_{t\to\infty} \frac{\mathbb{K}(t)}{t}$. We also define $\Omega = \sum_{i=1}^{n} \frac{c_i}{w_i}$.

We say a schedule is *periodic* if and only if there exists τ such that for any day d, the plants scheduled to be maintained are identical to the ones for day $d + \tau$. In a *strictly periodic* schedule, each plant labeled i is maintained exactly once every w_i consecutive days.

Properties. We prove the following properties.

PROPERTY 1. *For any feasible schedule, $\overline{K} \geq \Omega$.*

PROOF. We divide the schedule into blocks of length $b = lcm(w_1, \ldots, w_n)$, the least common multiple of $\{w_1, \ldots, w_n\}$. By the maintenance window constraint, plant i is scheduled to be maintained at least $\frac{b}{w_i}$ times in each block, and the average daily workload in each block is at least $\frac{\sum_{i=1}^{n} \frac{c_i b}{w_i}}{b} = \Omega$.

Now consider the $\frac{\mathbb{K}(t)}{t}$ sequence of this schedule. Suppose $t = pb + t'$, where $0 \leq t' < b$. $\frac{\mathbb{K}(t)}{t} \geq \frac{pb}{t}\Omega$. Since $\lim_{t\to\infty} \frac{pb}{t} = 1$, $\overline{\lim}_{t\to\infty} \frac{\mathbb{K}(t)}{t} \geq \Omega$. □

PROPERTY 2. *If there is a schedule for $(k, \overline{k}, (c_1, w_1), \ldots, (c_n, w_n))$, there exists a periodic schedule for $(k, \overline{k}', (c_1, w_1), \ldots, (c_n, w_n))$ for some number $\overline{k}'$.*

PROOF. Suppose there is a $(K_{max}, \overline{K})$ schedule X for instance $((c_1, w_1), \ldots, (c_n, w_n))$. We divide X into blocks of length $b = \max\{w_1, \ldots, w_n\}$. Since the number of types of blocks is finite, there exists a block appearing twice in X. Suppose $S, S_1, \ldots, S_m, S$ is part of X that contains two appearances of block S.

Consider schedule $S, S_1, \ldots, S_m, S, S_1, \ldots, S_m, S, \ldots$. Obviously, its largest daily workload is no more than $K_{max} \leq k$. Now we show the schedule also satisfies the maintenance window constraint for each plant. For any plant i, any w_i consecutive days in the schedule must be contained in two blocks that are also part of the original schedule. Hence, plant i is scheduled at least once in these w_i days. □

PROPERTY 3. *If there is a schedule for $(k, \overline{k}, (c_1, w_1), \ldots, (c_n, w_n))$, then for any $\varepsilon > 0$ there exists a periodic schedule for $(k, \overline{k} + \varepsilon, (c_1, w_1), \ldots, (c_n, w_n))$.*

PROOF. Let X be a schedule for $(k, \overline{k}, (c_1, w_1), \ldots, (c_n, w_n))$. We divide X into blocks of length $b = \max\{w_1, \ldots, w_n\}$. Since the number of types of blocks is finite, there must be a block that appears infinite number of times in X. Let S be such a block, and d_j be the last day of the jth appearance of S. We divide the schedule into supper-blocks, $\{0, \ldots, d_1\}$, $\{d_1 + 1, \ldots, d_2\}$, ..., denoted by $u_1, u_2, \ldots$. Let $\mathbb{K}(u_j)$ denote the total maintenance costs of plants in u_j, and $|u_j|$ denote the number of days in supper-block u_j.

Consider the subsequence $\{\frac{\mathbb{K}(d_j)}{d_j}\}, j = 1, 2, \ldots$ of sequence $\{\frac{\mathbb{K}(t)}{t}\}, t = 1, 2, \ldots$. Since

$$\overline{\lim_{j\to\infty}} \frac{\mathbb{K}(d_j)}{d_j} \leq \overline{\lim_{t\to\infty}} \frac{\mathbb{K}(t)}{t} \leq \overline{k} \text{ and } \lim_{m\to\infty} \frac{m}{m - |u_1|} = 1,$$

there exists m s.t. $\frac{\mathbb{K}(d_m)}{d_m} < \overline{k} + \varepsilon/4$ and $\frac{m}{m-|u_1|} < \frac{\overline{k}+\varepsilon/2}{\overline{k}+\varepsilon/4}$.

Since $\frac{\mathbb{K}(d_m)}{d_m} = \Sigma_{j=1}^{m}(\frac{|u_j|}{m} \cdot \frac{\mathbb{K}(u_j)}{|u_j|})$

$= \frac{|u_1|}{m}\frac{\mathbb{K}(u_1)}{|u_1|} + \frac{m-|u_1|}{m}(\Sigma_{j=2}^{m} \frac{|u_j|}{m-|u_1|} \cdot \frac{\mathbb{K}(u_j)}{|u_j|})$,

$$\Sigma_{j=2}^{m} \frac{|u_j|}{m - |u_1|} \cdot \frac{\mathbb{K}(u_j)}{|u_j|} < (\overline{k} + \varepsilon/4)\frac{m}{m - |u_1|} < \overline{k} + \varepsilon/2.$$

Note that the left-hand side is a weighted average number of $\frac{c(u_j)}{|u_j|}$, so there exist j' such that $\frac{c(u_{j'})}{|u_{j'}|} < \overline{k} + \varepsilon/2$.

We now construct a periodic schedule $S, u_{j'}, u_{j'}, \ldots$. By Property 2, this is a $(K_{max}, \overline{K})$ schedule. We only need to prove $\overline{K} < \overline{k} + \varepsilon$.

We can find a large enough number m', such that the contribution of days not in the whole period of $\frac{\mathbb{K}(m')}{m'}$ is smaller than $\varepsilon/2$. Since in the period u_j, $\frac{\mathbb{K}(u_{j'})}{|u_{j'}|} < \overline{k} + \varepsilon/2$, the conclusion follows. □

Hardness. We first prove the Gardener's problem is NP-hard by a reduction from the Bin Packing problem. Then we give a stronger NP-hardness result for the unweighted Gardener's problem, where the maintenance costs for all plants are 1. The idea is to let $k' = \Omega$ and force any feasible schedule to be strictly periodic.

THEOREM 3.1. *The Gardener's problem is NP-hard.*

PROOF. Let $I = (w, k, c_1, \ldots, c_n)$ be an instance of the NP-hard Bin Packing problem, which asks whether n items (plants) with weights $c_1, \ldots, c_n$ can be packed into w bins, each having capacity k.

We reduce I to $I' = (k, \overline{k} = k, (w, c_1), \ldots, (w, c_n))$, an instance of the Gardener's problem.

If I has a solution, there is a solution for I', where the plants scheduled on day $d + wt$ ($t \in \mathbf{N}$) are exactly the plants in the dth bin.

If I' has a feasible solution, then each plant is scheduled at least once in the first w days. Since the maximum daily workload is k, we can pack all plants into w bins. □

THEOREM 3.2. *The unweighted Gardener's problem is NP-hard.*

PROOF. An NP-hardness proof [1] is given for a variant of the Gardener's problem, where there is no $\overline{k}$ specification, $k = 1$, and the task is to find a strictly periodic schedule.

Given I as an instance of the unweighted Gardener's problem, we ask whether there is a schedule with $k = 1$ and $\overline{k} = \Omega$. If there is a strictly periodic schedule for I, since its average workload is Ω, it is also a feasible schedule for I for the Gardener's problem.

We say a segment of length $lcm(w_1, \ldots, w_n)$ of a schedule is *good* if in a concatenation of two such segments, the distance between each two consecutive appearances of any plant i is exactly $w_i - 1$. Obviously, if there is a good segment, there is a strictly periodic solution.

If there is a solution for I, by Property 3, for any ε, there is a $(k, \overline{k} + \varepsilon)$ periodic solution. Here we need to do some minor modification for the proof of property 3. Instead of having $b = \max\{w_1, \ldots, w_n\}$, we let b be $lcm(w_1, \ldots, w_n)$ multiplied by $4 \cdot \max\{w_1, \ldots, w_n\}$. We take $\varepsilon < \frac{1}{b}$. Suppose $\tau = \lambda b$ is the period of this periodic solution. The average workload of one period is less than $\Omega + \varepsilon$.

In a *strict period*, in which all plants are strictly periodically scheduled, the average workload is Ω. Now, we can have at most $\varepsilon\tau = \lambda$ more plants than the strict period. We divide one period into λ blocks of length b. There must be one block that gets at most one more plant than the strict period for this block. Suppose this plant is the ith one. In this block, there are at most w_i pairs of consecutive appearances of plant i having distance less than $w_i - 1$. We divide this block into $4 \cdot \max\{w_1, \ldots, w_n\}$ segments. These w_i pairs will make at most $2w_i$ segments not good. There is at least one good segment left in this block, hence there is a strictly periodic solution. □

Approximation. Since it is hard to decide whether there is a solution satisfying k, it is hard to give an approximation algorithm optimizing $\overline{K}$ while satisfying k. In the following we show a simple algorithm that gives a 4-approximation for the Gardener's problem minimizing $\overline{K}$.

LEMMA 3.1. *Given $k, ((c_1, w_1), \ldots, (c_n, w_n))$, an instance of the Gardener's problem minimizing $\overline{K}$, where each w_i has a form of 2^q and each c_i, $0 < c_i \leq 1$, has the form of 2^{-q}. There exists a schedule with workload $\Omega + 2$.*

PROOF. Suppose $2^s = \max\{w_1, \ldots, w_n\}$, and P_s is the set of all plants with window size of 2^s. Let C_s denote the summation of the maintenance costs of plants in P_s.

We first pack all plants in P_s into $\lceil C_s \rceil$ bins of size 1. Then we merge every two bins into a plant with window size of 2^{s-1}. Thus, we can replace all plants in P_s by $\lceil \lceil C_s \rceil / 2 \rceil$ plants with window size of 2^{s-1} to get a new instance I'. If I' has a $(K_{max}, \overline{K})$ solution, we can get a $(K_{max}, \overline{K})$ solution for the original instance I. $\Omega_{I'} - \Omega_I = (2\lceil \lceil C_s \rceil / 2 \rceil - C_s)2^{-s} \leq 2^{1-s}$.

For I', the largest window size is 2^{s-1}. We repeat the same process until we get an instance I'' with the largest window size of 1. $\Omega_{I''} - \Omega_I \leq \sum_{i=s}^{1} 2^{1-i} \leq 2$. Since I'' has a solution with workload $\Omega_{I''}$, there is a solution for I with workload no more than $\Omega_I + 2$. □

THEOREM 3.3. *There exists a polynomial time algorithm for the Gardener's problem that gives a schedule with workload no more than $4\Omega + 2$. The average daily workload $\overline{K}$ of this schedule is also no more than $4\Omega + 2$.*

PROOF. Given an instance I for the Gardener's problem, we get a new instance I' by rounding each w_i to the maximum number in the form of 2^q that is no more than w_i, and each cost c_i to the minimum number in the form of 2^q that is no less than c_i.

Since each c_i and each $\frac{1}{w_i}$ are enlarged no more than twice, by the definition of Ω, $\Omega_{I'} \leq 4\Omega_I$. Applying lemma 3.1 to I' gives us a schedule with workload no more than $4\Omega + 2$. □

Note that Ω is a lower bound for both workload and average daily workload by Property 1.

4. THE OPTIMIZATION PROBLEM

In this section, we present a multicriteria formulation for the Gardener's optimization problem, for which we propose a heuristic scheduling algorithm MMEDD.

Measures and Objectives. In the Gardener's problem, we have two measures: tardiness and workload, from which we can derive various objective functions, or criteria.

For a plant i, there are many tardiness values, one for each maintenance day. Let $T_i(d)$ be its tardiness of maintenance at day d. We define $T_i(d) = 0$ if plant i is not maintained on day d. Then for plant i, the largest tardiness $T_i = \lim_{t\to\infty} \max_{1\leq d\leq t}\{T_i(d)\}$, and the average tardiness $\overline{T}_i = \lim_{t\to\infty} \frac{1}{t}\sum_{d=1}^{t} T_i(d)$. Then, the largest tardiness T_{max} and average tardiness $\overline{T}$ over all plants are

$$T_{max} = \max_{1\leq i\leq n}\{T_i\} \qquad \overline{T} = \frac{1}{n}\sum_{i=1}^{n}\overline{T_i}$$

We use $K(d)$ to denote the workload of the gardener at day d, then the largest daily workload K_{max} and average daily workload $\overline{K}$ over all days are

$$K_{max} = \lim_{t\to\infty}\max_{1\leq d\leq t}\{K(d)\} \qquad \overline{K} = \lim_{t\to\infty}\frac{1}{t}\sum_{d=1}^{t} K(d)$$

Minimization of T_{max}, $\overline{T}$, K_{max} and $\overline{K}$ are all reasonable objectives. For web information monitoring applications, usually there is a hard constraint given for K_{max}, and minimizing $\overline{T}$ and $\overline{K}$ is much desirable.

Problem Definition. A gardener has n plants to maintain. Each plant labeled i needs to be maintained at least once every w_i days and each maintenance costs c_i workload. The gardener has a daily workload capacity of k. Let $\overline{K}$ be the average daily workload over all days. Let $\overline{T}$ be the average tardiness over all days and all plants. Assuming all plants were maintained yesterday, find a feasible schedule starting from today until forever so that $\overline{K}$ and $\overline{L}$ are minimized.

We use a tuple, $(k, (c_1, w_1), \ldots, (c_n, w_n))$, to denote an instance of the above Gardener's optimization problem. The problem is NP-hard. A multicriteria problem is NP-hard if the optimization of a single criterion is NP-hard. In the following we propose greedy heuristics.

EDD. A simple greedy algorithm EDD (Earliest Due Day first) provides optimal solutions for many non-maintenance scheduling problems minimizing the maximum tardiness [6]. In EDD, jobs (plants) are ordered by non-decreasing order of due dates and scheduled in that order.

However, if used in maintenance scheduling, EDD would "exhaust" the gardener. Consider a simple instance with $n = 2$, $w_1 = w_2 = 365$, $c_1 = c_2 = 1$, and $k = 2$. Optimally, the gardener only needs to work one day a year if he maintains plants 1 and 2 on their due days. However, with EDD, the gardener would maintain the two plants every day and work 365 days a year! Instead of early completion, maintenance scheduling concerns long-term in-time maintenance. Jobs will not be completed. Each maintenance of a job simply generates a new due day for the same job. EDD would end up generating too many due days unnecessarily.

MEDD. MEDD is a maintenance version of EDD that minimizes tardiness while concerning workload. Algorithm 1 presents the pseudocode of MEDD.

Let L_i denote the current *lateness* of plant i, i.e., the number of days after its current due day. The difference between lateness and tardiness is that lateness can be a negative value, indicating the due day is in the future. Obviously, "earliest due day first" is the same as "largest lateness first".

In the algorithm, we first calculate the lateness for each plant based on its current due day (line 1). Then, we sort all plants in non-increasing order of lateness (line 2). We use the smallest cost first criterion to break ties in order to maintain more plants. Then, we choose as many as possible (satisfying the k constraint) the top ranked plants with lateness ≥ 0 to maintain. $k_{residue}$ is the remaining capacity that can be used to maintain some plants due in the future (line 3). Let w_{max} be the maximum window size (line 4). Assign to c the estimated total maintenance cost in the next w_{max} days assuming all plants are maintained on their due days (line 5). Then assign to *overload* the difference of c and the accumulated capacity over w_{max} days (line 6).

At this point, the non-negative *overload* represents the workload *absolutely* needed in the next m_{max} days in order to maintain all plants in time. Thus we would not waste any workload if we spend $min(k_{residue}, overload)$ workload to relieve some maintenance pressure for the future. So we choose some plants (line 7, to be explained shortly) that use up $min(k_{residue}, overload) \leq k$ to maintain (line 8). Plants in M_1 are due earlier or today. Plants in M_2 are due in the future but the workload is well spent. They together form the set M of plants scheduled to maintain (line 9). We assume the plants in M are always maintained as scheduled, thus we update the due days for all of them (line 10).

Algorithm 1 *MEDD*

Input: $(k, (c_1, w_1), \ldots, (c_n, w_n))$
Output: M: a set of labels for plants scheduled to be maintained
1: calculate L_i for each plant i;
2: sort the plants in non-increasing order of lateness, breaking ties smallest cost first and then largest window first;
3: choose the set of top plants M_1 such that $k_{residue} = k - \sum_{i \in M_1} c_i \geq 0$ and $\forall i \in M_1, L_i \geq 0$;
4: $w_{max} \leftarrow max(w_1, w_2, \ldots, w_n)$;
5: $c \leftarrow$ estimate the total maintenance cost in the next w_{max} days assuming all plants are maintained on their due days;
6: $overload \leftarrow \max(0, c - w_{max} * k)$;
7: sort the plants in non-decreasing order of wasteness;
8: choose the set of top plants M_2 such that $\sum_{i \in M_2} c_i \leq min(overload, k_{residue})$;
9: $M \leftarrow M_1 \cup M_2$;
10: update the due day for each plant $i \in M$;

In line 7, it may seem more natural to use the same criteria as in line 2, largest lateness first, to select plants. However, there is a fundamental difference. In line 2, we want to choose some plants that are due already. In line 7, we want to choose some plants that are due in the future. An early maintenance incurs wasteness of workload, where the *wasteness* of plant i is defined as $\frac{L_i}{w_i} c_i$. Here L_i represents the wasted days. It is normalized by w_i and weighted c_i.

MMEDD. MEDD still emphasizes minimizing tardiness as in EDD but uses workload wisely. It returns a good schedule with respect to the given daily capacity k.

MMEDD, the multicriteria version of MEDD, tries to find other good alternative schedules with respect to some $k' < k$. We can simply repeatedly generate some $k' < k$ and feed $(k', (c_1, w_1), \ldots, (c_n, w_n))$ to MEDD.

5. CONCLUSION

We focused on theoretical aspects of the Gardener's problem. The implemented OLAP system with built-in schedulers is available at http://dmlab.cs.txstate.edu/scube/, as well as a more complete technical report with experiments.

Although our study is in the context of web information monitoring, our results and approaches can be applied to applications related to information monitoring in general, e.g., materialized view maintenance in data warehouses.

6. REFERENCES

[1] A. Bar-Noy, R. Bhatia, J. Naor, and B. Schieber. Minimizing service and operation costs of periodic scheduling. *Mathematics of Operations Research*, 27(3):518–544, 2002.

[2] A. Bar-Noy, R. E. Ladner, and T. Tamir. Windows scheduling as a restricted version of bin packing. *ACM Transactions on Algorithms*, 3(3), 2007.

[3] B. E. Brewington and G. Cybenko. Keeping up with the changing web. *IEEE Computer*, 33:52–58, 2000.

[4] J. Cho and H. Garcia-Molina. Synchronizing a database to improve freshness. In *SIGMOD*, 2000.

[5] E. Coffman, Z. Liu, and P. Weber. Optimal robot scheduling for web search engines. *Journal of Scheduling*, (1):15–29, 1998.

[6] C. S. David R. Karger and J. Wein. *Scheduling Algorithms*. Algorithms and Theory of Computation Handbook, 1998.

[7] A. Doan et al. Community information management. *IEEE Data Engineering Bulletin*, 29(1):64–72, 2006.

[8] J. Edwards, K. McCurley, and J. Tomlin. An adaptive model for optimizing performance of an incremental web crawler. In *WWW*, 2001.

[9] M. Ehrgott. *Multicriteria optimization*. Springer, 2005.

[10] M. Pinedo. *Scheduling - Theory, Algorithms, and Systems*. Prentice Hall, 1995.

[11] V. T'kindt and J.-C. Billaut. *Multicriteria Scheduling*. Springer, 2006.