On How to Merge Sorted Lists Coming from Different Web Search Tools

Ronald R. Yager[1] and Vladik Kreinovich[2]

[1]Machine Intelligence Institute
Iona College
715 North Avenue
New Rochelle, NY 10801-18903, USA
email yager@panix.com

[2]Department of Computer Science
University of Texas at El Paso
El Paso, TX 79968, USA
email vladik@cs.utep.edu

Abstract

Different web search tools often complement each other. So, if we want to have a good coverage of all relevant web items, a reasonable strategy is to use different search tools and then merge the resulting lists. How to merge them? In this paper, we describe reasonable axioms for the merging procedure and describe all mergings that satisfy these reasonable axioms.

1 Introduction

Currently, the World Wide Web contains a great amount of (often poorly organized) information. To get the information about a certain topic, we can use one of the known search tools. Several successful tools are used on the web, and the reason why they all successfully co-exist is that they are based on radically different principles and as a result, these tools complement each other. For example, while one tool may be better to find the most visited pages that contain the search words, other tools may complement these many-hit pages by adding other webpages, which may not be visited that frequently but in which the searched words are truly central.

Usually, if we want a good coverage of all relevant websites, it is reasonable to use *several* search tools, and then *combine* (merge) the resulting lists. The question is: What is the best way of combining these lists?

The reason why this question is important is because each of the lists usually comes *sorted*: first comes the webpage that is, according to the search tool, the most relevant; then comes the page that is next in relevance, etc., and the last item on the list is (according to this search tool) the least relevant of those found. We would like to merge the lists coming from different search tools into a single list that is similarly sorted, so that this list would start with the (possibly) most relevant item, then continue with second most relevant, etc. So, in mathematical terms, the question is: how to combine merge sorted lists?

In this paper, we consider the situation when the merged lists do not overlap (i.e., that lists coming from different search tools only complement each other).

> This situation happens, e.g., when different search tools cover different sources. Suppose, for example, that we are interested in news about the Thailand economy. To get the full picture, we go to the websites of different periodicals that cover economic news, such as e.g., Financial Times, Economist, Wall Street Journal, etc. Each of these websites has its own search tool, which returns the webpages that contain the corresponding articles. As a result, for each periodical, we have a list of relevant articles published in this periodical. Different lists describe articles published in different periodicals, so these lists do not overlap.

What information can we use for this merger?

- Definitely, we have an *order* for each list.
- This order, usually, originates from the *numerical* values that this search tool assigns to each found item. It may seem, at first glance, that we can use these values too in our merging. However, since different search tools are based on completely different ideas and principles, there is *no* known *way to* meaningfully *compare the numerical values* generated by different search tools.

So, order is the only information about each list that we can use, and the problem becomes: *How to merge sorted lists?*

2 Towards mathematical formulation of the problem

We want numerical values. Let us denote the number of lists that we want to merge by n. The lists themselves can be denoted by $L_1, \ldots, L_n$. For each j from 1 to n, let us denote by N_j the number of items in j-th list.

A natural way of merging these n lists into a single sorted list is to assign, to each element of each of the lists, a value v, and then sort all $N_1 + N_2 + \ldots + N_n$

elements in the increasing order of this value. So, the question is: how can we determine this value v for each item?

We need a function of two variables. An item is uniquely characterized by two parameters:

- the number j of the list L_j from which this item comes and
- the order i of the given item in the corresponding list L_j (first item is No. 1, second item is No. 2, etc.).

Hence, the desired value v must be uniquely determined by the values of these two parameters j and i.

The only information we use to produce a merged list consists of the lists themselves. Therefore, if we have two lists L_j and L_k of equal length (i.e., two lists with the same number of elements $N_j = N_k$), we have no reason to believe that one of these lists is better than the other one. So, if two lists L_j and L_k have the same number of elements, then i-th element of the list L_j should be assigned the same value as i-th element of the list L_k.

> This argument in the same spirit as the *principle of insufficient reason* that lies in the foundations of statistics (especially Bayesian statistics) and of decision making [4, 3]. This principle, first introduced by Jacob Bernoulli, and then popularized by Pierre Simon de Laplace in his famous treatise on probabilities [2], states that if there is no evidence leading one to believe that one event is more likely than another, then the events should be judged equally probable. Similarly, since we have no evidence that one of the two lists with the same number of elements is better than another, we assign equal values of v to similarly placed elements of these two lists.

In other words, the dependence on j can only appear through dependence on N_j. Hence, the value v must be uniquely determined by i and N_j, i.e., we should have $v = v(i, N_j)$.

The question is: which function of two variables i and N_j should we choose?

First requirement: the function $v(i, N)$ should preserve the equality between drops in relevance. Our first requirement is as follows: For each list, the only information that we consider is its order. This order describes the decreasing relevance of the items on the list: the first item is of the largest relevance to the query, the second item is of smaller relevance, etc.

Thus, for example, the "drop in relevance" between the first and the third elements of the list is larger than the drop in relevance between the first and second elements, because:

- the drop in relevance from 1st to 3rd elements occurs in *two* steps (a drop from 1st to 2nd, and then further drop from 2nd to 3rd), while
- the drop in relevance from 1st to 2nd elements consists of only *one* step.

On the other hand, if we compare two drops in relevance that take only one step, e.g.:

- the drop in relevance from the 1st to the 2nd element, and
- the drop in relevance from the 2nd to the 3rd element,

then there is no reason to believe that one of these "drops in relevance" is larger than the other one, and therefore, following the same "principle of insufficient reason" as we used before, we can conclude that these two "drops of relevance" are equal.

It is reasonable to require that this equality of drops in relevance should be preserved by the value-assigning function $v(i, N)$ (which defines the merged list). We introduce the function $v(i, N)$ to describe the ordering in the merged list: first, we will list the item with the smallest value of v, then the item with the second smallest value of v, etc. Our objective is to describe the items in the merged list in the order of their relevance to the query: the most relevant item should come first, the second relevant should second, etc. Thus, the value $v(i, N_j)$ describes the *relevance* of item i from j-th list in the merged list. Hence, the informal notion of "drop in relevance" from item i to item $i+1$ can be naturally formalized as the *difference* between the corresponding values of the function v (which describe their relevance), i.e., as the difference $v(i+1, N) - v(i, N)$.

Within this formalization, the requirement that the function v should preserve the equality between the drops in relevance means that the above-defined "drop in relevance" (difference) from 1st to 2nd items must be the same as the "drop in relevance" difference from the 2nd to the 3rd items, etc. In other words, this requirement means that $v(2, N) - v(1, N) = v(3, N) - v(2, N) = \ldots = v(i+1, N) - v(i, N) = \ldots$

Similarly, the "drop in relevance" from the 1st to the 3rd item must be same as the drop in relevance from the 2nd to the 4th, from the 3rd to the 5th. etc., i.e., $v(3, N) - v(1, N) = v(4, N) - v(2, N) = \ldots = v(i+2, N) - v(i, N) = \ldots$

In general, if $k - i = k' - i'$, i.e., if the in original list, the "drop in relevance" from i-th to k-th items is the same as the drop in relevance from i'-th to k'-th items, then in the resulting list, we should also have equal drops in relevance, i.e., the drop in relevance $v(k, N) - v(i, N)$ from i-th to k-th items must be equal to the drop in relevance $v(k', N) - v(i', N)$ from i'-th item to the k'-th item.

Since the cut-off that led to each list is, usually, reasonably arbitrary, the function $v(i, N)$ should be defined for $i > N$ as well. In the above text, we proposed to define the values $v(i, N_j)$ for every element i in j-th list. Since j-th list has N_j elements, we want to define this expression for all values $i = 1, \ldots, N_j$. Thus, we need a function $v(i, N)$ to be defined for all possible pairs of positive integers for which $i \leq N$.

The restriction $i \leq N$ is indeed reasonable if each of the search tools produces exactly N_j items and nothing else. In reality, the situation is more complicated: each search tool produces much more than what it shows us, and then cuts off the resulting long list. In other words:

- first, a search tool produces a long list of possibly relevant items, and
- then, it cuts off the tail of the original list, keeping only the most relevant items.

The cut-off is usually done according to some heuristic (and reasonably arbitrary) criterion, and often, some items that were cut off are not much less relevant than the items that remained in the list.

With the possibility of this too-harsh cut-off in mind, it makes sense to consider the possibility that some items, which were initially cut off from the search tool lists, will be later on added to these lists, and thus, these additional items will be available for merging.

Let us describe the consequences of this possibility in mathematical terms. Originally, we had a list with N_j items numbers $1, 2, \ldots, N_j$. To this original list, we add further elements, i.e., elements numbers $N_j + 1$, $N_j + 2$, etc. We now have additional elements to merge, so we need to assign the values v to these additional elements.

- From the *theoretical* viewpoint, the new situation does not require any new definitions. Namely, since we added elements to j-th list, its length increases, from the original value N_j to some new value $N'_j > N_j$. So, all we have to do is:
  - *re-calculate* the values $v(i, N'_j)$ for *old* items $i = 1, \ldots, N_j$, and
  - *calculate* the values $v(i, N'_j)$ for *new* items $i = N_j + 1, N_j + 2, \ldots, N'_j$.

  In both cases, we only need to consider the function $v(i, N)$ for $i \leq N$.
- However, from the *practical* viewpoint, the above theoretical procedure does not make much sense. Indeed, suppose that we only added *one* element. Then:

- Adding an extra element does not change the relative relevance of all other elements, so, at first glance, we should preserve the values of old elements, and calculate a single new value: the value of the new element (so that we will be able to find its proper place in the merged list).
- However, according to the above procedure, we still need to re-calculate *all* the values.

It is desirable to avoid this waste of computation time, and be able to simply add one number (corresponding to the new element) to the results of previous calculations. In other words, it is desirable to add the value $v(N_j+1, N_j)$ that corresponds to the first additional element.

Similarly, it is reasonable to consider the values $v(N_j+2, N_j)$, $v(N_j+3, N_j)$, etc., which correspond to second, third, etc. elements. In general, it is, thus, reasonable to consider the values $v(i, N)$ not only for $i \le N$, but for $i > N$ as well.

In other words, it makes sense to consider the function $v(i, N)$ for *arbitrary* positive integers i and N.

Second requirement: the merged order should not change if we simply change the granularity. In order to describe this second requirement, we must elaborate a little bit on a fine point of web search tools. This point is best illustrated by an example.

One of the authors (V.K.) is a co-maintainer of the Interval Computations website (http://cs.utep.edu/interval-comp/main.html). This website consists of the main page, a dozen pages that this page points to, and several dozen pages that these pages, in their turn, point to. Some web search tools, if asked to provide all the information relevant to interval computations, produce only the main page of this website, while other web search tools generate the list of *all* (or at least many) pages from this website.

This example can be reformulated in more abstract terms: different search tools use different levels of *granularity*: where one tool returns a single page, another tool may return several of them. Even for the same web search algorithm, it is usually possible to tune this algorithm in such a way that the resulting tool may exhibit different levels of granularity.

A natural requirement is that if we change the level of granularity for all the tools, then the merged order should not change. How can we express this requirement in mathematical terms?

Let us assume that we have tuned all search algorithms so that they now produce several pages where they used to produce a single one. A natural way to describe this "tuning" in quantitative terms is by describing the average increase t in the number of pages, i.e., by the average number of new pages per single old page.

Due to this increase, a list that originally contained N_j items would now contain approximately $t \cdot N_j$ items, and, since each item on the old list corresponds to t items in the new list, the item number i in the old list has, in the new list, number $\approx t \cdot i$.

If each old page correspond to *exactly* t new pages, then instead of approximate equality, we get an exact equality: item number i (of N_j) on the *old* j-th list (i.e., on the list obtained by j-th web search tool *before* tuning) is item number $t \cdot i$ (of $t \cdot N_j$) on the new j-th list (i.e., on the list produced by j-th search tool *after* tuning). In this case, the requirement that the tuning should not change the order of the merged list can be formulated as follows:

> If in the original merged list, item # i from the list of N was placed before item # i' from the list of N' (i.e., if $v(i, N) < v(i', N')$), then in the new list, the corresponding items should be placed in the exact same order, i.e., $v(t \cdot i, t \cdot N) < v(t \cdot i', t \cdot N')$.

In real life, t is only the *average* increase in the number of pages, not the *exact* one. Since t is an average increase, the values $t \cdot i$ and $t \cdot N$ need not be integers. It is, therefore, natural to consider non-integer (*real*) values of i and N as well, and thus, to consider the function $v(i, N)$ for arbitrary *real* values i and N.

This is not just a mathematical trick: The possibility of changing the granularity level leads to a natural interpretation of such non-integer values of i and N. This interpretation is best described by the example of the *opposite* transformation, from an extended list to the shortened one, a transformation in which instead of an *increase*, we get a *decrease* in the number of pages (i.e., in which $t < 1$). For example, if two old pages correspond to a single new one, then $t = 0.5$, and so, old item No. 2 correspond to new item No. 1, old item No. 4 is new No. 2, etc. Items Nos. 1, 3, 5, etc. in the old list are simply absent from the new list, in the sense that their information is still available, but not directly anymore: to get the information from old item No. 1, we have to go to new item No. 1, and then click on its subpage, etc. To describe the "indirect" presence of these items in the new list, we can assign to them non-integer numbers $t \cdot i$: e.g., old No. 1 is now No. 0.5, old No. 3 is now new No. 1.5, etc.

In view of this interpretation, it is reasonable to require that the function $v(i, N)$ be defined for arbitrary rational numbers i and N, and that for arbitrary i, N, i', N', and t, $v(i, N) < v(i', N')$ should imply $v(t \cdot i, t \cdot N) < v(t \cdot i', t \cdot N')$.

This motivation justifies only the use of *rational* numbers i and N. However, most mathematical functions, methods, and formulas use arbitrary *real* numbers. It is therefore convenient to use arbitrary *real* values i and N instead of only rational ones. By using real numbers, we do not add anything, because an arbitrary real number can be approximated, within an arbitrary accuracy, by a rational one, so, from any practical purposes, whether we consider only rational numbers or arbitrary real values is rather irrelevant (for example, inside a computer, every real number is represented as a rational number anyway). Since the only reason for our using irrational values is, thus, to approximate rational ones, it is natural to require that for close inputs, the function should have close values, i.e., that this function $v(i, N)$ should be *continuous*.

Thus, we arrive at the following definition:

3 Definition and the main result

Definition. *We say that a continuous real-valued function $v(i, N)$ of two positive real variables describes merger if it satisfies the following two conditions:*

- *for every i, k, i', k', and N, if $k - i = k' - i'$, then $v(k, N) - v(i, N) = v(k', N) - v(i', N)$;*
- *for every i, i', N, N', and t, $v(i, N) < v(i', N')$ if and only if $v(t \cdot i, t \cdot N) < v(t \cdot i', t \cdot N')$.*

Comment. The first condition means that equally spaced items remain equally spaced; the second condition says that the resulting order should not not change if we change granularity level.

Theorem. *A function $v(i, N)$ describes merger if and only if it has the form $v(i, N) = N^\alpha \cdot (c_1 \cdot i + c_2 \cdot N) + c_0$ for some real numbers α, c_0, c_1, and c_2.*

Comment. Since we are only using the values $v(i, N)$ to compare different items, we can safely set $c_0 = 0$, because adding an arbitrary number c_0 to all values $v(i, N)$ does not change the relative order of these values.

Similarly, multiplying all values $v(i, N)$ by a positive constant does not change the order, so, we can safely assume that $c_1 = 1$.

After these two simplifications, we get an expression $v(i, N) = N^\alpha \cdot (i + c_2 \cdot N)$ with only two parameters: α and c_2.

In [5, 6], we considered a formula $v(i, N) = i + c_2 \cdot N$ which corresponds to $\alpha = 0$, and we showed that already this simplified formula enables us, by finding an appropriate value of c_2, to produce a merger that expert would consider reasonable. We hope that if we are allowed not only this parameter c_2, but also the additional parameter α, then the resulting merged list that would be even closer to how an expert would manually merge the corresponding lists.

4 Proof

Let us first fix N and consider $v(i, N)$ as a function of i. From the first condition (that equally spaced items remain equally spaced) we conclude, in particular, that $v(2, N) - v(1, N) = v(3, N) - v(2, N) = \ldots = v(i + 1, N) - v(i, N) = \ldots$ for all positive integers i. If we denote this common difference by $b(N)$, we thus conclude that:

- $v(2,N) - v(1,N) = b(N)$, hence
$$v(2,N) = v(1,N) + b(N);$$
- $v(3,N) - v(2,N) = b(N)$, hence
$$v(3,N) = v(2,N) + b(N) = v(1,N) + 2b(N);$$
- ...
- $v(i,N) - v(i-1,N) = b(N)$ and hence,
$$v(i,N) = v(i-1,N) + b(N) = v(1,N) + (i-1)\cdot b(N),$$
- ...

Thus, for all integer i, we have $v(i,N) = v(1,N) + (i-1)\cdot b(N)$. This formula can be further simplified if we denote $v(1,N) - b(N)$ by $a(N)$, then this formula takes the form
$$v(i,N) = a(N) + i \cdot b(N). \tag{1}$$

We have shown that this formula holds for all *integer* values of i. Let us show that it also holds for "half-integer" values, i.e., values of the type $i = p/2$, where p is an integer. Indeed, let us show, that this formula is true for
$$i = k + \frac{1}{2}.$$
We know that $v(k+1,N) - v(k,N) = b(N)$. We also know that, since
$$(k+1) - \left(k + \frac{1}{2}\right) = \left(k + \frac{1}{2}\right) - k,$$
we have
$$v\left(k+1,N\right) - v\left(k + \frac{1}{2}, N\right) = v\left(k + \frac{1}{2}, N\right) - v\left(k,N\right).$$
The sum of these two equal values is equal to
$$\left(v\left(k+1,N\right) - v\left(k + \frac{1}{2}, N\right)\right) + \left(v\left(k + \frac{1}{2}, N\right) - v\left(k,N\right)\right) =$$
$$v\left(k+1,N\right) - v\left(k,N\right) = b\left(N\right),$$
and thus, each of these terms is equal to $(1/2)\cdot b(N)$. Thus,

$$v\left(k+\frac{1}{2},N\right)=v\left(k,N\right)+\left(v\left(k+\frac{1}{2},N\right)-v\left(k,N\right)\right)=$$
$$\left(a\left(N\right)+k\cdot b\left(N\right)\right)+\frac{1}{2}\cdot b\left(N\right)=a\left(N\right)+\left(k+\frac{1}{2}\right)\cdot b\left(N\right).$$

A similar proof can be repeated for all half-integer values i.

Similarly, we can prove that the formula (1) holds for all "quarter-integer" values i, i.e., values of the type $p/2^2$, also for values of the type $p/2^3$, $p/2^4$, and $p/2^q$ for arbitrary p and q. Since we can approximate an arbitrary real number i by such values, and we have assumed that the function $v(i,N)$ is continuous, we thus conclude that the formula (1) is true for all i and N.

In view of formula (1), to describe the function $v(i,N)$, it is sufficient to describe the functions $a(N)$ and $b(N)$. To obtain such description, let us use the second condition from our definition. Namely, according to this condition, for every i, i', N, N', and t, $a(N)+b(N)\cdot i<a(N')+b(N')\cdot i'$ if and only if $a(t\cdot N)+b(t\cdot N)\cdot t\cdot i<a(t\cdot N')+b(t\cdot N')\cdot t\cdot i'$.

So, if

$$a(N)+b(N)\cdot i=a(N')+b(N')\cdot i', \tag{2}$$

then:

- we cannot have $a(t\cdot N)+b(t\cdot N)\cdot t\cdot i<a(t\cdot N')+b(t\cdot N')\cdot t\cdot i'$, because otherwise, we would have $a(N)+b(N)\cdot i<a(N')+b(N')\cdot i'$;
- we cannot have $a(t\cdot N)+b(t\cdot N)\cdot t\cdot i>a(t\cdot N')+b(t\cdot N')\cdot t\cdot i'$, because otherwise, we would have $a(N)+b(N)\cdot i>a(N')+b(N')\cdot i'$;

and therefore, we must have

$$a(t\cdot N)+b(t\cdot N)\cdot t\cdot i=a(t\cdot N')+b(t\cdot N')\cdot t\cdot i'. \tag{3}$$

From the equation (2), we can uniquely determine i':

$$i'=\frac{a(N)-a(N')}{b(N')}+\frac{b(N)}{b(N')}\cdot i. \tag{4}$$

From equation (3), we similarly conclude that

$$i'=\frac{a(t\cdot N)-a(t\cdot N')}{t\cdot b(N')}+\frac{b(t\cdot N)}{b(t\cdot N')}\cdot i. \tag{5}$$

Thus, the second condition means that for every i, N, N', and t, the right-hand sides of the equations (4) and (5) must coincide, i.e., that we must have

$$\frac{a(N)-a(N')}{b(N')}+\frac{b(N)}{b(N')}\cdot i=\frac{a(t\cdot N)-a(t\cdot N')}{t\cdot b(N')}+\frac{b(t\cdot N)}{b(t\cdot N')}\cdot i. \tag{6}$$

Both sides are linear functions of i. Since the two linear functions are always equal, their coefficients at i should be equal, and their free terms are equal. Hence, we get the following two equalities:

$$\frac{b(N)}{b(N')} = \frac{b(t \cdot N)}{b(t \cdot N')}; \tag{7}$$

$$\frac{a(N) - a(N')}{b(N')} = \frac{a(t \cdot N) - a(t \cdot N')}{t \cdot b(t \cdot N')}. \tag{8}$$

The equation (7) contains only one of the unknown functions ($b(N)$), while (8) contains both of them. So, for simplicity, let us start with analyzing equation (7). Let us start by exploiting particular cases of this equation. Substituting $N' = 1$ into equation (7), we conclude that

$$\frac{b(N)}{b(1)} = \frac{b(t \cdot N)}{b(t)},$$

and hence, that

$$b(t \cdot N) = \frac{b(t) \cdot b(N)}{b(1)}. \qquad (9).$$

This functional equation can be further simplified if we introduce a new unknown function $c(N) = b(N)/b(1)$. If we substitute $b(N) = c(N) \cdot b(1)$ into the equation (9) and divide both sides of this equation by $b(1)$, we conclude that $c(t \cdot N) = c(t) \cdot c(N)$. This functional equation dates back to Cauchy, and it is well known that every continuous solution to this equation is of the form $c(N) = N^\alpha$ for some real number α (see, e.g., [1]). Thus, $b(N) = b(1) \cdot N^\alpha$. This is exactly the coefficient at i that we want to obtain in our Theorem, with the only difference that in that formulation, we used c_1 instead of $b(1)$. So, if we denote $b(1)$ by c_1, we get the desired expression

$$b(N) = c_1 \cdot N^\alpha. \tag{10}$$

For this $b(N)$, formula (8) take the following form:

$$\frac{a(N) - a(N')}{c_1 \cdot (N')^\alpha} = \frac{a(t \cdot N) - a(t \cdot N')}{t \cdot c_1 \cdot (t \cdot N')^\alpha}.$$

Multiplying both sides of this equation by $c_1 \cdot (N')^\alpha \cdot t^{1+\alpha}$, we conclude that

$$a(t \cdot N) - a(t \cdot N') = t^{1+\alpha} \cdot (a(N) - a(N')).$$

Substituting $N' = 1$, and moving the term $a(t \cdot 1) = a(t)$ into the right-hand side, we conclude that

$$a(t \cdot N) = a(t) + t^{1+\alpha} \cdot a(N) - t^{1+\alpha} \cdot a(1). \tag{11}$$

"Swapping" t and N and taking into consideration that $t \cdot N = N \cdot t$, we can conclude that

$$a(t \cdot N) = a(N \cdot t) = a(N) + N^{1+\alpha} \cdot a(t) - N^{1+\alpha} \cdot a(1). \tag{12}$$

Since the left-hand sides of equations (11) and (12) coincide, their right-hand side must also coincide for arbitrary i and N:

$$a(t) + t^{1+\alpha} \cdot a(N) - t^{1+\alpha} \cdot a(1) = a(N) + N^{1+\alpha} \cdot a(t) - N^{1+\alpha} \cdot a(1). \tag{13}$$

If we move terms that contain $a(N)$ to the left-hand side and all other terms to the right-hand side, we conclude that

$$(t^{1+\alpha} - 1) \cdot a(N) = t^{1+\alpha} \cdot a(1) + N^{1+\alpha} \cdot a(t) - N^{1+\alpha} \cdot a(1) - a(t). \tag{14}$$

This equality must be true for all t. Let us pick one value of t, e.g., $t = 2$ (one can easily check that any other value of t would lead to the same result). Substituting $t = 2$ into the equality (14), we conclude that

$$(2^{1+\alpha} - 1) \cdot a(N) = 2^{1+\alpha} \cdot a(1) + N^{1+\alpha} \cdot a(2) - N^{1+\alpha} \cdot a(1) - a(2), \tag{15}$$

i.e., that

$$a(N) = c_2 \cdot N^{1+\alpha} + c_0, \tag{16}$$

where we denoted

$$c_2 = \frac{a(2) - a(1)}{2^{1+\alpha} - 1}$$

and

$$c_0 = \frac{2^{1+\alpha} \cdot a(1) - a(2)}{2^{1+\alpha} - 1}.$$

Substituting expressions (16) and (10) into the formula (1), we get the desired expression for the merging function. The theorem is proven.

Acknowledgments. This work was partially supported by NASA under cooperative agreement NCCW-0089, by NSF grants No. DUE-9750858 and EEC-9322370, and by the Future Aerospace Science and Technology Program (FAST) Center for Structural Integrity of Aerospace Systems, effort sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant number F49620-95-1-0518.

The authors are thankful to all the participants of the American Association for Artificial Intelligence (AAAI) Symposium on Frontiers in Soft Computing and Decision Systems (MIT, Boston, MA, November 8–10, 1997), especially to Larry Medsker, Lotfi A. Zadeh, and Hans-Jürgen Zimmermann, and to all participants of the Knowledge Representation Seminar (El Paso, TX), especially to Michael Gelfond, for valuable discussions.

References

[1] J. Aczel, *Lectures on Functional Equations and Their Applications*, Academic Press, N.Y.-London, 1966.

[2] R. S. de Laplace, *Essai philosophique sure les probabilités*, Paris, 1814 (1st edition); English translation of the last author's edition: *A philosophical essay on probabilities*, Wiley, N.Y., 1917; reprinted by Dover, N.Y., 1952.

[3] D. R. Luce and H. Raiffa, *Games and Decisions, Introduction and critical survey*, J. Wiley & Sons, N.Y., 1957, reprinted by Dover, N.Y., 1989.

[4] L. J. Savage, *The foundations of statistics*, Wiley, N.Y., 1954; reprinted by Dover, N.Y., 1972.

[5] R. R. Yager and A. Rybalov, "On the fusion of documents from multiple collection information retrieval systems," *Journal of the American Society for Information Sciences* (to appear).

[6] R. R. Yager and A. Rybalov, "Retrieving documents from multiple information sources," in: D. Mancini, M. Squillante, and A. Ventre, *New Trends in Fuzzy Systems*, World Scientific Publishers, Singapore (to appear).