Editor: Isabel Beichl, isabel.beichl@nist.gov

IT'S BOUND TO BE RIGHT

By Isabel Beichl and Francis Sullivan

WE RECENTLY USED A MONTE CARLO METHOD FOR ELIMINATING DOUBLE COUNTING. THE SAME IDEA IS USEFUL FOR CHARACTERIZING THE "STATE" OF AN IMAGE AUTOMATICALLY, WHICH IS PARTICULARLY INTEResting because photographs and digital images have become so commonplace. Automatic detection of change of state is suddenly desirable. What we do is compute a multiplicative factor ρ, which we use to tighten the Bonferroni bound on the size of a union of sets, assuming we already know the individual sizes of the sets. We then obtain a new bound that is more accurate and useful.

We will explain how to use this new bound to estimate the number of edges in a triangulation if we are given the tetrahedra (in 3D). We will also explain how to eliminate duplication in complicated data sets. Finally, we will explain how to use the same idea to characterize images.

First, let's examine the Bonferroni bound and how to compute ρ.

The Bonferroni bound

Suppose we have a union of two sets A and B

$$S = A \cup B$$

and suppose that we know the size of A, $|A|$, and the size of B, $|B|$. We also assume that we can sample uniformly from A or B and that we can easily determine if an element is in A, B, or both. We would like to estimate the size of S, $|S|$. We know that

$$|A \cup B| = |A| + |B| - |A \cap B| \leq |A| + |B|.$$

This inequality's right side is the Bonferroni bound. It's not a very good estimate of $|A \cup B|$ because the intersection $|A \cap B|$ can be considerable. For more than two sets, the Bonferroni bound is

$$|A_1 \cup A_2 \cup \ldots \cup A_n| \leq |A_1| + |A_2| + \ldots + |A_n|.$$

We will estimate a number, $\rho < 1$, so that

$$|A \cup B| \approx \rho * (|A| + |B|),$$

which would give us the desired estimate. For more complicated systems, of course,

$$|A_1 \cup A_2 \cup \ldots \cup A_n| = \rho * (|A_1| + |A_2| + \ldots + |A_n|).$$

We will show how this is done with just two sets, and we comment on how it works for any number of sets.

Computing ρ

We define two probabilities, P_A and P_B,

$$P_A = \frac{|A|}{|A| + |B|}, P_B = \frac{|B|}{|A| + |B|}$$

such that

$$P_A + P_B = 1.$$

Here is the algorithm for ρ. Choose one of set A or B—set A with probability P_A, set B with probability P_B. (We call this *stabbing*. Briefly, generate a random number $0 < r < 1$. If $r < P_A$, choose A; otherwise, choose B. It works for any number of sets $A_1 \ldots A_n$ at the same time.) If you choose A, then choose an element of A uniforml, similarly for B. The probability of choosing any one element of A will be

$$\frac{1}{|A|};$$

the probability of choosing any one element of B will be

$$\frac{1}{|B|}.$$

What is the probability in this procedure for choosing one particular element of $S = A \cup B$? There are three disjoint

 1521-9615/02/$17.00 © 2002 IEEE

possibilities: the element is in $A \setminus B$, $B \setminus A$, or $A \cap B$. If the element is in the first possibility, A but not B, then the probability of selecting it was

$$\frac{|A|}{|A|+|B|} * \frac{1}{|A|} = \frac{1}{|A|+|B|}.$$

Similarly if the element was in B but not A, we would get the same probability of being chosen—namely, $1/(|A|+|B|)$. If the element were in the intersection, then it would be twice as likely to have been chosen, and the probability would be $2/(|A|+|B|)$.

So here's how we estimate ρ. Select an element of $A \cup B$ with the above sampling procedure. Let k_i be the number of sets (1 or 2—that is, either in one of A or B, or in both A and B) that the element is in:

$$\rho = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{k_i},$$

where n is the number of samples taken. So, we claim that

$$\rho * (|A| + |B|) \approx |A \cup B|.$$

For N sets $A_1 \ldots A_N$, select set A_i with probability

$$|A_i| / \sum_{j=1}^{N} |A_j|$$

and then select a sample from the A_i chosen.

Why does this work?

Consider an example. (For a complete explanation, see George S. Fishman's book *Monte Carlo: Concepts, Algorithms and Applications* [Springer-Verlag, 1996].) Suppose that we have two sets,

$$A = \{a, b, c, d\} \qquad \text{and} \qquad B = \{c, d, e, f\}$$

Note that $|A \cup B| = 6$, but $|A| + |B| = 8$. We assume as usual that it is easy to tell if an element is in $A \cap B$. Now we do the following Monte Carlo experiment. Choose uniformly either A or B. This means that A is chosen with probability

$$\mathcal{P}_A = \frac{|A|}{|A|+|B|}$$

and B is chosen with probability

$$\mathcal{P}_B = \frac{|B|}{|A|+|B|}.$$

Next, we choose an element uniformly within the set chosen. So, if we choose A, any particular element in it is chosen with probability

$$\frac{1}{|A|},$$

and similarly for B. In either case, multiplying the "choose a set" probability by the "choose an element" probability, we get

$$\frac{1}{|A|+|B|}.$$

Call this the *uniform probability*, $\mathcal{P}$.

Now, let's calculate the true total probability of choosing, say, f, by this process. First, we'd have to choose B, and then in B, we'd have to choose f. So, the probability of choosing f is

$$\frac{|B|}{|A|+|B|} * \frac{1}{|B|} = \frac{1}{|A|+|B|}.$$

However, the *true* total probability of choosing, say, d, is

$$\left(\frac{|A|}{|A|+|B|} * \frac{1}{|A|}\right) + \left(\frac{|B|}{|A|+|B|} * \frac{1}{|B|}\right) = \frac{2}{|A|+|B|}.$$

We'll choose elements from A or B by the procedure described earlier and record the inverse of the true probability of selecting an element as just described. We do this by first choosing and then noting if the element chosen is in one or two sets. (Recall that we assume this check is easy to do.) So, each sample will be either

$$\frac{|A|+|B|}{1} \qquad \text{or} \qquad \frac{|A|+|B|}{2}.$$

What will the mean of the samples be? Clearly something smaller than $|A| + |B|$, because every term has this factor multiplied by either 1 or 1/2. So the mean will be

$$(|A| + |B|) * \rho,$$

where ρ is the mean of the 1's and 1/2's. In our example, what will ρ turn out to be in the long run? It will be

$$\rho = \mathcal{P}(\{a, b, e, f\}) * 1 + \mathcal{P}(\{c, d\}) * \frac{1}{2}$$
$$= \frac{4}{8} * 1 + \frac{4}{8} * \frac{1}{2} = \frac{6}{8}$$

```
clear g
clear ts

r=rand(npt,2);           %npt=number of points to triangulate

x=r(:,1);
y=r(:,2);

tri=delaunay(x,y);       %tri is the set of triangles
sz=size(tri);
stz= sz(1);              %stz is the number or triangles

for s=1:samp
            tr=floor(stz*rand)+1;       %choose triangle at random
            vert=tri(tr,:);             %vertices or tr

            pt1=floor(rand*3)+1;
            v1=vert(pt1);
            vert=setdiff(vert,v1);      %choose random vertex and remove from list

            pt2=floor(rand*2)+1;
            v2=vert(pt2);               %choose second vertex

            t1=tsearch(x,y,tri,x(v1),y(v1)); %triangles that contain v1
            t2=tsearch(x,y,tri,x(v2),y(v2));

     g(s)=length(intersect(t1,t2));     %number of containing edge (v1, v2)
     ts(s)=tr;                          %triangle used

end

     rho=mean(ones(1,samp)./g)          %adjustment: Number of edges ~

                                        %rho*3*sz(1)
```

Figure 1. Matlab code for 2D triangulation followed by an estimate of the number of edges `=rho3*(#triangles)`, **where** `rho` **is calculated with the method described in this article.**

where $\mathcal{P}(S)$ equals the probability of selecting an element of set S.

Finally, notice that if we multiply ρ by the Bonferroni bound, we get what we were looking for:

$$\rho * (|A| + |B|) = \frac{6}{8} * 8 = 6 = |A \cup B|.$$

More generally, if n is the number of samples taken, and k_i is the number of sets containing sample number i, then

$$\rho = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{k_i}.$$

(In our previous two-set example, k_i was 2 if i was in $A \cap B$; otherwise, it was 1.) We let $\mathcal{P}(\sigma)$ be the probability that element σ of $A \cup B$ is chosen by the previous method, and k_σ is the number of sets that element σ is in. So, in the long run, ρ equals

$$\sum_{\sigma \in A \cup B} \mathcal{P}(\sigma) \frac{1}{k_\sigma} = \sum_{\sigma \in A \cup B} \frac{k_\sigma}{|A| + |B|} * \frac{1}{k_\sigma} = \sum_{\sigma \in A \cup B} \frac{1}{|A| + |B|} = \frac{|A \cup B|}{|A| + |B|}.$$

So, of course,

$$\rho * (|A| + |B|) = |A \cup B|.$$

You might have noticed that this is really an application of importance sampling.

Example: Counting edges in a triangulation

Suppose we have generated a large, 3D grid made up of tetrahedra. In most cases, we'll know how many tetrahedra are in the grid, but other questions might not be so easy to answer. For example, how many edges are there? Each tetrahedron has six, so there could be as many as six times the number of tetrahedra. However, the tetrahedra touch one another, so edges are shared. For random data, the average edge is used in six tetrahedra, so to a very rough approximation, the number of edges is equal to the number of tetrahedra. This can't be exact, of course. Consider the case of two tetrahedra meeting at one face.

How can we be more accurate without doing more work? We could just choose a random edge and see how many tetrahedra T it is in. The average of $1/T$ would then be the ρ associated with the upper bound of 6*(# tetrahedra). In 2D, where a triangulation would give triangles, we could estimate the number of edges by using the upper bound of 3*(#triangles) and then using this method to calculate a ρ. We could guess that ρ will be close to 1/2 because almost every edge appears in two triangles (see Figure 1). Only the boundary triangles do not.

Another example

Suppose that we had lots of images (say, 10,000) and that they were processed by some algorithms, A_1, A_2, A_3, ..., each of which looks for some feature F_1, F_2, F_3, ... (Assume that these programs already exist and that not every image has a feature.) If the feature is present, the image is put into a database. The problem is that they might all be put in the same database haphazardly, so after processing, the database has 12,000 entries.

Two questions:

1. How big should the database really be?
2. If we redo the database to be the right size, how can we determine the probability of an image that has several features at the same time? In other words, can we tell when an image is "different" than the usual one?

The first question means find the ρ_{rs} such that ρ_{rs} * 12,000 = *RightSize*. To do it, choose an image i at random and determine (by running the feature algorithms) how many features it has. This is $g(i)$. If we take n samples, we have

$$\rho_{rs} = \frac{1}{n}\sum_i \frac{1}{g(i)}.$$

To answer the second question, we use the fact that if $p(F_i)$ is the probability of feature i being present (equal to the fraction of images that have feature i), then the probability that F_i is not present in an image, which we'll call $p(F_i^c)$, is

$$p\left(F_i^c\right) = 1 - p(F_i).$$

The probability of, say, two features present at the same time is

$$p(F_i \cap F_j) = 1 - p\left(F_i^c \cup F_j^c\right).$$

To finish, we compute a number ρ_c such that

$$p\left(F_i^c \cup F_j^c\right) = \rho_c\left(p(F_i^c) + p(F_j^c)\right).$$

This is the probability of two features being in the same image. So you see, now you can get an estimate of the probability of having several features.

Imagine using this technique to build a table of probabilities for states of a system, given a massive number of photographs. In building the table, you would need to know what state the photograph was in. For each set of features $\{F_i\}$, you would calculate the probability of having combinations of these features. Once you have the table, someone could come along with another photograph. You could then count the features and look up the photograph in the table to see the probability of it being in a certain state.

You can find several real-world examples with more detail on using these bounds to classify the state of an image in Daniel Q. Naiman and Carey Priebe's paper, "Computing Scan Statistic p-Values Using Importance Sampling, with Applications to Genetics and Medical Image Analysis" (*J. Computational and Graphical Statistics*, vol. 10, no. 2, June 2001, pp. 296–328). CiSE

Isabel Beichl is the department editor for Computing Prescriptions. She is also a mathematician at the National Institute of Standards and Technology. Contact her at isabel.beichl@nist.gov.

Francis Sullivan is the editor in chief of *Computing in Science & Engineering*. He is also the director of the IDA Center for Computing Sciences. Contact him at fran@super.org.