Non-Blood Relationship Searches

by Wesley Johnston

How is Fidel Castro related to Napoleon Bonaparte? In 1965, I imagined having the ultimate genealogical database and answering such a question. Nineteen years later, in 1984, personal computing had finally caught up a bit with my idea. So I published an article in *GC* (May, 1984) on doing blood relationship searches in genealogical databases. Now 19 more years have gone by, and the best genealogical database products all have robust blood relationship searches. But they still can't answer my 1965 question. It is time to raise the bar—good genealogical database software must include non-blood relationships. This article will explain how.

The Inquiry Framework

I will use the same framework that I used in 1984. We can ask these questions in a relationship search:

1. Is X related to Y? Yes or no?
2. What is the relationship?
3. Who are the people linking them together?

For non-blood searches, questions 1 and 3 are what matter most. The second question may have an answer that is too difficult to understand so it may be omitted from early versions. Later versions can use a complexity threshold, so that relatively simple descriptions of relationships could be generated, such as "2nd great-grandmother of wife of 5th cousin 3 times removed." But many will not be that simple. So for now, let's put question 2 aside.

Key Thoughts

Two people can be connected only if each one is in at least one family. So the key to the search is to match families, rather than individuals. In the search, an individual is treated as a relationship between families, as well as the other way around.

The Process

The search expands outward one level at a time, first from Person X and then from Person Y at each expansion level. The expansion ends when a connecting relationship is found or when one side finds no more families that can be expanded (i.e., when there is no relationship). It is important to remember that an expansion does not just go up to ancestors or down to children. Expansions are like ripples, spreading out from a stone dropped in a pond—they go out in all directions—so an expansion level is not the same thing as a generation level.

Here are the steps that we will cycle through at each expansion level.

1. **Expand X:** For each unexpanded family in the previous level's Expansion Set for X, create an entry in this level's Expansion Set by expanding any not-yet-expanded individuals in the family.
2. **Check for Match in X:** Check to see if any of the families in the newest Expansion Set for X have already been expanded on the Y side. If so, then X and Y are related, and you can then display the relationship(s) and stop.

It is time to raise the bar—good genealogical database software must include non-blood relationships.

3. **Expand Y:** Repeat step 1 for Y.
4. **Check for Match in Y:** Repeat step 2 for Y.
5. **Next Level:** Return to step 1 for the next higher expansion level.

A walk-through of a complete example follows the section on data structures.

The Data Structures

The right data structure makes it easier to do the search.

1. RIN-Name: a lookup index, to go from an individual's number (RIN) to their names, used only after the search for the final report.
2. Pairs (for the X side and the Y side) of arrays of on/off flags, to keep track of which persons and families have been expanded.
3. Family-Person: an array that lists all people who are in a given family.
4. Person-Family: an array that lists all families to which a given person belongs.
5. Expansion Sets for both X and Y, one pair of sets per expansion level.

RIN-Name almost certainly exists in most genealogical database software packages. The arrays of flags are simple to construct. The other arrays deserve a more detailed look.

When we find our way back to our ancestors in small villages, we find a high degree of intermarriage between families that went on intermarrying for centuries, creating a highly intricate network of relationships.

Family-Person and Person-Family

Each record in Family-Person is indexed by family number (MRIN) and contains the RINs of all those people who constitute the family. The RINs are signed numbers, positive or negative, so that we can indicate whether the person is a spouse (positive) or a child (negative) in the family.

Person-Family is the inverse of Family-Person. Person-Family is indexed by RIN and contains the MRINs of all families to which the person belongs. The MRINs are signed numbers: positive if the person is a spouse in the family, negative if the person is a child in the family.

Expansion Sets

The Expansion Sets are the most complicated data structures. The Expansion Sets allow us to determine whether there is a matching family at some point. They also allow us to backtrack (once one or more matches are found), so that we can report the results of the search.

An Expansion Set for a given side (X or Y) and for a given expansion level contains quintuples of five pieces of information:

1. MRin1: the number of the generating family
2. Role1: the person's role in the generating family (Boolean: True=Spouse, False=Child)
3. XRin: the person's RIN
4. MRin2: the number of the generated family number
5. Role2: the person's role in the generating family (Boolean: True=Spouse, False=Child)

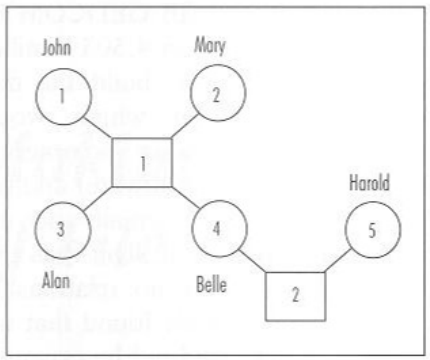
For the first level, since there is no prior set, the generating MRin1 will be zero. When the generating MRin1 is zero, the value of the generating Role1 is ignored.

When we go forward, during the expansion, we use the lower level's MRin2 to generate the next level's MRin1. When we have discovered a match and are backtracking to answer Question 3, we use the higher level's MRin1 to find the same family in the lower level's MRin2.

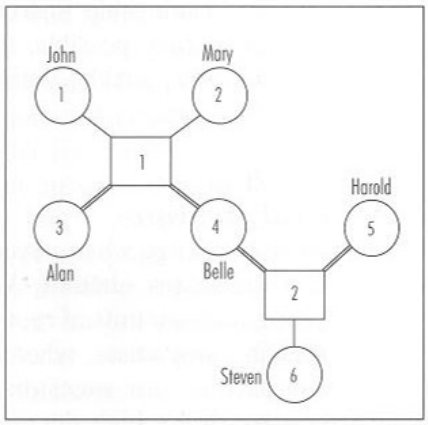
For example, consider the tuple (271, TRUE, 14386, 1187, FALSE). The person's RIN is the middle number: 14386. This tuple tells us that person 14386 is a child (the "False") in family 1187 and a spouse (the "True") in family 271. It also tells us that family 271 was generated in the prior level, so that there is at least one tuple in the prior level that has 271 for its MRin2 value.

A Complete Example

In a complete search, we specify both an X person and a Y person. So let's search for the relationship between Harold (X) and Alan (Y) in the simple database in the diagram. The RINs are in the circles, and the MRINs in the boxes.



Now we go through the five steps for each expansion level. Here is the expansion the following process will generate:



Expansion Level 1–Step 1–Expand X: We expand Harold (RIN 5), finding all families in which he is a member. He is in only one family (family 2), and he is a spouse in that family. So Expansion Set X1 is {(MRin1=0, Role1=F, XRin=5, MRin2=2, Role2=T)}. We set the X-side flag arrays to show that we have expanded RIN 5 and MRIN 2.

Expansion Level 1–Step 2–Check for Match in X: Since we have not created any Expansion Set for Y yet, none of set X1's families will have also been found on the Y-side. So we go on to the next step.

Expansion Level 1–Step 3–Expand Y: We expand Alan (RIN 3). He is in only one family (family 1), and he is a child in that family. So Expansion Set Y1 is {(0, T, 3, 1, F)}. We set the Y-side flag arrays to show that we have expanded RIN 3 and MRIN 1.

Expansion Level 1–Step 4–Check for Match in Y: For each MRin2 in the set Y1, we look at the X-side MRIN array, to see if that family has already been expanded on the X side. None have been expanded. So there is no match, and our search goes on to the next step.

Expansion Level 1–Step 5–Next Level: Increase the expansion level number to 2, and go through the five steps again.

Expansion Level 2–Step 1–Expand X1: We take each tuple (there is only one) in Expansion Set X1, and we find all of the members of that tuple's generated family (MRin2m which is 2). We try to expand any people who have not yet been expanded.

Family 2 is that of Harold and Belle and Steven, and Belle and Steven have not yet been expanded. So we try to expand Belle (MRIN 4). And we can expand her, since she also appears in family 1, where she is a child. When we try to expand Steven, we cannot, since he does not appear in any other family.

So the Expansion Set X2 is {(2, T, 4, 1, F)}, which says that Belle is spouse in family 2 and a child in family 1. We set the X-side flag arrays to show that we have expanded RIN 4 and MRIN 1.

Expansion Level 2–Step 2–Check for Match in X1: For each MRin2 in the set X2, we look at the Y-side MRIN array, to see if that family has already been expanded on the Y side. And we find that family 1 has been expanded on the Y side. So we have found a relationship path that connects Harold and Alan, and we have our answer to Question 1: Yes, they are related.

Answering Question 3 requires that we now step backward through the Expansion Sets on each side. The X2 tuple that led us to the match enables us to print out the first line of the backtracking: "Belle is a child in the matched family."

Now we compare the generating family (MRin1) in the X2 tuple (family 1) with the generated family (MRin2) in the prior level for X (which is X1). We know that it will be in at least one tuple of X1, since that was how it came to exist in X2. We find the X1 tuple that allows us to print out the second (and final) line on the X side backtrack:

Non-blood relationship searches are not only theoretically possible, they work—and work very quickly—without combinatorial explosion.

"Harold (Person X) is a spouse in the family (2) in which Belle is a spouse."

Then we switch over to the Y side and find the tuple in Y1 that has the matched family (1) in MRin2. We can then print out the only line on the Y side backtrack: "Alan is a child in the matched family."

The discovered relationship path can also be shown graphically.

MRin1	Role1	XRin	MRin2	Role2
0	F	5	2	T
0	F	3	1	F
2	T	4	1	F

Multiple Relationships

The example does not show what happens when a second or third relationship is found, but it is exactly the same as for the first relationship. We search all tuples in the expansion level at which any match was discovered, and we print out the backtracking separately for each match we find.

NBR Search Version 1.0

When I searched for genealogical database software that had non-blood relationship searches, I found none. I also found perceptions that such a search was not possible, that it is too complicated, or that it would lead to a "combinatorial explosion," so that while it was theoretically possible, no practical non-blood relationship search could ever be done.

The time has long since come for this non-blood relationship search capability to be a standard feature of any genealogical database software.

So, as I did for blood relationship search articles in 1984, I have gone beyond the theory that I have spelled out above. I have written a non-blood relationship search to put the theory into practice. I wrote it in as simple and generally available a language as possible—Microsoft *Excel's Visual Basic*. And I am placing the code on the Web until at least

http://members.aol.com/wwjohnston/nbrsearch.htm/

31 December 2004, so that anyone with *Excel* can search a GEDCOM file for non-blood relationships.

The results of this simple program were surprising, even to me. On a 500 MHz Pentium II, with 256MB of memory, I searched an 8MB GEDCOM file of 13,425 individuals in 4,503 families. It took 22 seconds to build the data structures (much of which would already be in place in commercial genealogical database software) and less than one second to actually do any search that I tried on it. This was true even when there was no relationship, since the search quickly found that one side or the other could not be expanded further. No search required more than 12 expansion levels. The largest expansion set in any search had 224 tuples.

So non-blood relationship searches are not only theoretically possible, they work—and work very quickly, without combinatorial explosion.

So Let's See It in Genealogical Software

We are now in an age when massive family history databases abound. Web sites are making vast amounts of records available. And in many cases, when we find our way back to our ancestors in small villages, we find a high degree of intermarriage between families that went on intermarrying for centuries, creating a highly intricate network of relationships. So we have and will continue to have a growing need for better tools that allow us to explore and understand all the complexities of our larger and richer databases.

Frankly, I am surprised that non-blood relationship search has not been implemented in the 19 years since I wrote my first article. The time has long since come for this non-blood relationship search capability to be a standard feature of any genealogical database software that seeks to be considered of good quality. So let's see it happen.

Wesley Johnston began genealogical research at age seven. His interests led to two Master's degrees, in mathematics and history. The mathematics led to computers, and he has long recognized the potential of using computers for family and local history. He does limited research now due to a health-related problem and cannot accept e-mail correspondence.