Parenting Partnering: Analysis

It is clear that whenever one parent has an activity, the other parent must care for the baby throughout that activity. But how should Cameron and Jamie divide up their remaining time after all activities are covered?

Small dataset

In the Small dataset, there are at most two activities, so we can use casework:

- If only one parent has an activity, an optimal solution is to have the other parent care for the baby in a continuous 720 hour block that includes that activity. Then the parent with the activity can handle the other 720 hours. So only 2 exchanges are required.
- If both parents have one activity each, the same idea as above works. It is always possible to choose some way to divide the 24-hour cycle exactly in half, such that Cameron's activity falls completely within Jamie's half and vice versa. So the answer is again 2.
- If one parent (without loss of generality, we'll say it's Cameron) has two activities, then they divide the 24-hour cycle such that there is a gap between activity 1 and activity 2, and a gap between activity 2 and activity 1. Jamie has to cover both activities. If one of these gaps is empty (which can occur if the activities are back-to-back) or can be completely filled in by Jamie, then the split the day in half strategy works again and the answer is 2. But if the activities are too far apart and/or too long, Jamie may not have enough remaining time to fill in either gap; in that case, both gaps must contain a switch from Jamie to Cameron and back, so the answer is 4. (We will explain later how we can fill in such gaps without adding more exchanges than we need to, regardless of how much time each parent contributes to filling in the gap.)

It does not take much code to implement the above arguments. However, care must be taken to handle midnight correctly when calculating the lengths of the gaps in the third case.

Large dataset

Let's consider a list of the activities, sorted by time. To gracefully handle the midnight transition and the time before the first activity / after the last activity, we will add a copy of the first activity to the end of the list. In this list, each interval is surrounded by two activities. Some intervals may be instantaneous transitions between activities that are back-to-back (let's call these "empty"); the other intervals represent all of the non-activity time. Each of these other intervals is either surrounded by activities covered by different parents, or by activities covered by the same parent.

We have no decisions to make about the empty intervals, but we must count up the exchanges they add. What about the different-parent intervals? For each, we will have to add some time from one or both parents to cover the interval; we can always do this optimally by starting with the time (if any) from the parent covering the activity on the left, and ending with the time (if any) from the parent covering the activity on the right. This strategy always leaves us with just one exchange. We can do no better than that (since the two different parents guaranteed that there would be at least one exchange), and we have no reason to do worse than that. So we do not need to worry about the different-parent intervals at all! We can assume that whatever time we have available (from one or both parents) can fill them up without creating any new exchanges.

Same-parent intervals require some more thought. If we can fill one in with time from the parent who is covering the endpoint activities, we can avoid an exchange, whereas if we include any time at all from the other parent, we will add *two* more exchanges. (We can avoid adding more than two by putting the time from the other parent in in one continuous chunk.) We may not be able to avoid the latter, depending on how much available time each parent has left, but we should fill as many intervals with "matching" time as we can, to minimize the number of new exchanges we create. Each interval contributes equally to the number of possible exchanges, but the intervals may have different lengths, and longer ones take up more of a parent's available time. So our best strategy is to sort the list of Cameron-bounded intervals and greedily fill in the shortest ones with Cameron time until we do not have enough Cameron time left to fill in the shortest remaining interval. Then, we do the same for the Jamie-bounded intervals. For every interval we fail to fill in this way, we add two exchanges to our total.

After we have handled the same-parent intervals, we are done and our current total number of exchanges is our answer. Whatever remaining time we have from one or both parents can safely go into the different-parent intervals without creating any new exchanges. As explained above, we do not need to think about the details; we leave those as an exercise for Cameron and Jamie.

This method is O(**N** log **N**) (because of the required sorting operations) and is easily fast enough for the Large dataset.