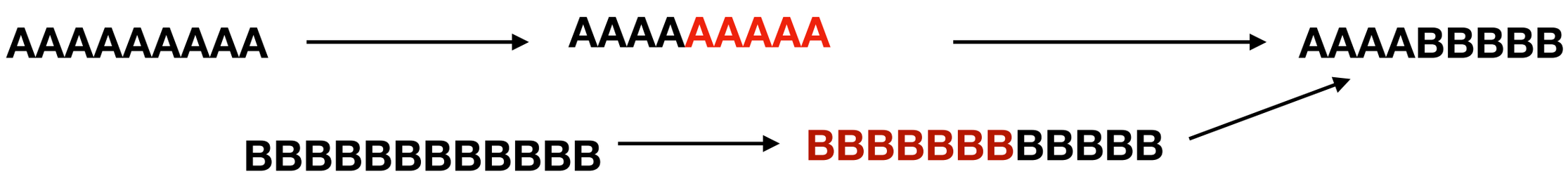
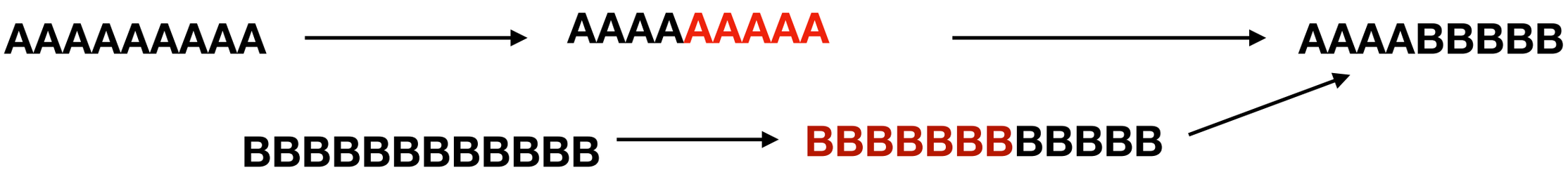
Microhomology-directed scenario model proposal



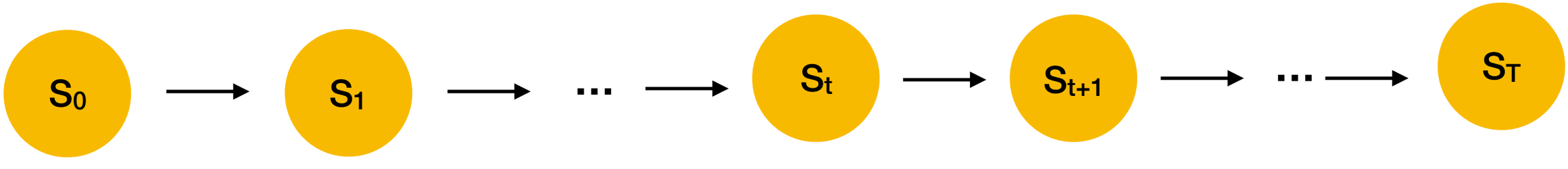
The problem:
Given two sequences and a known number of nucleotides that will be trimmed off, we want to learn the order of the steps in which trimming happens.
We hypothesize that the order matters since the transitional partially trimmed subsequences will interact via microhomology



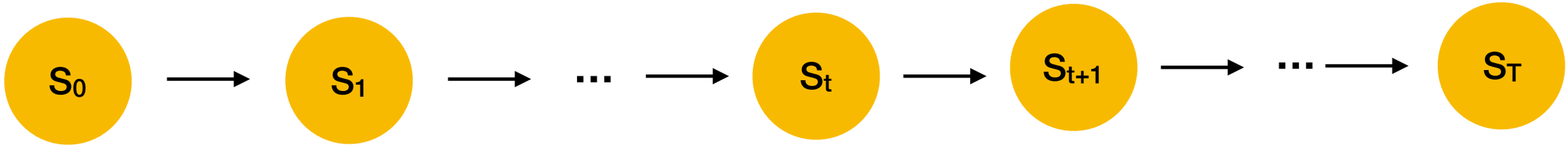
The problem:
Given two sequences and a known number of nucleotides that will be trimmed off, we want to learn the order of the steps in which trimming happens.
We hypothesize that the order matters since the transitional partially trimmed subsequences will interact via microhomology

Therefore, we suspect the system works by making trims that promote microhomology

Since we are interested in the order of single nucleotides trims, we model this with sequence states.

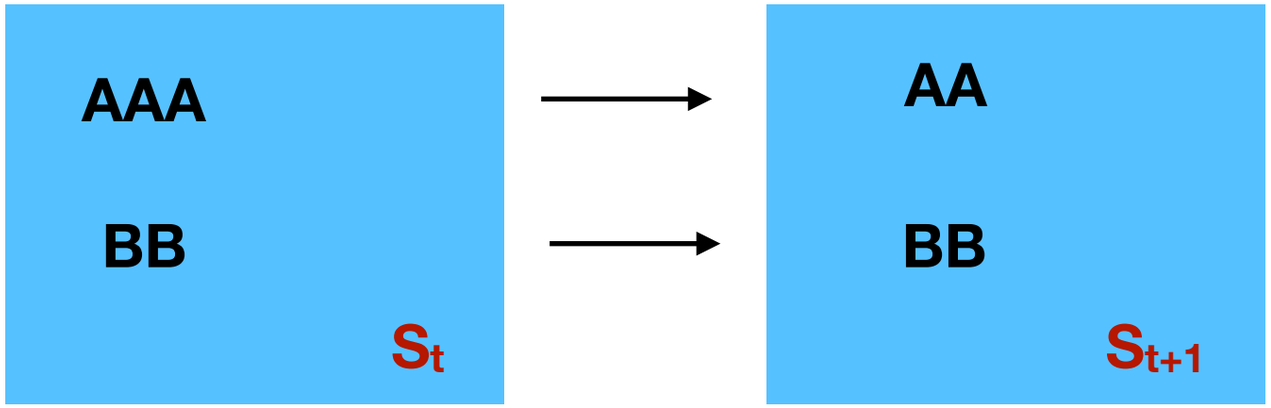


Since we are interested in the order of single nucleotides trims, we model this with sequence states.



Each state S_t refers to the collection of subsequences at that moment in time t

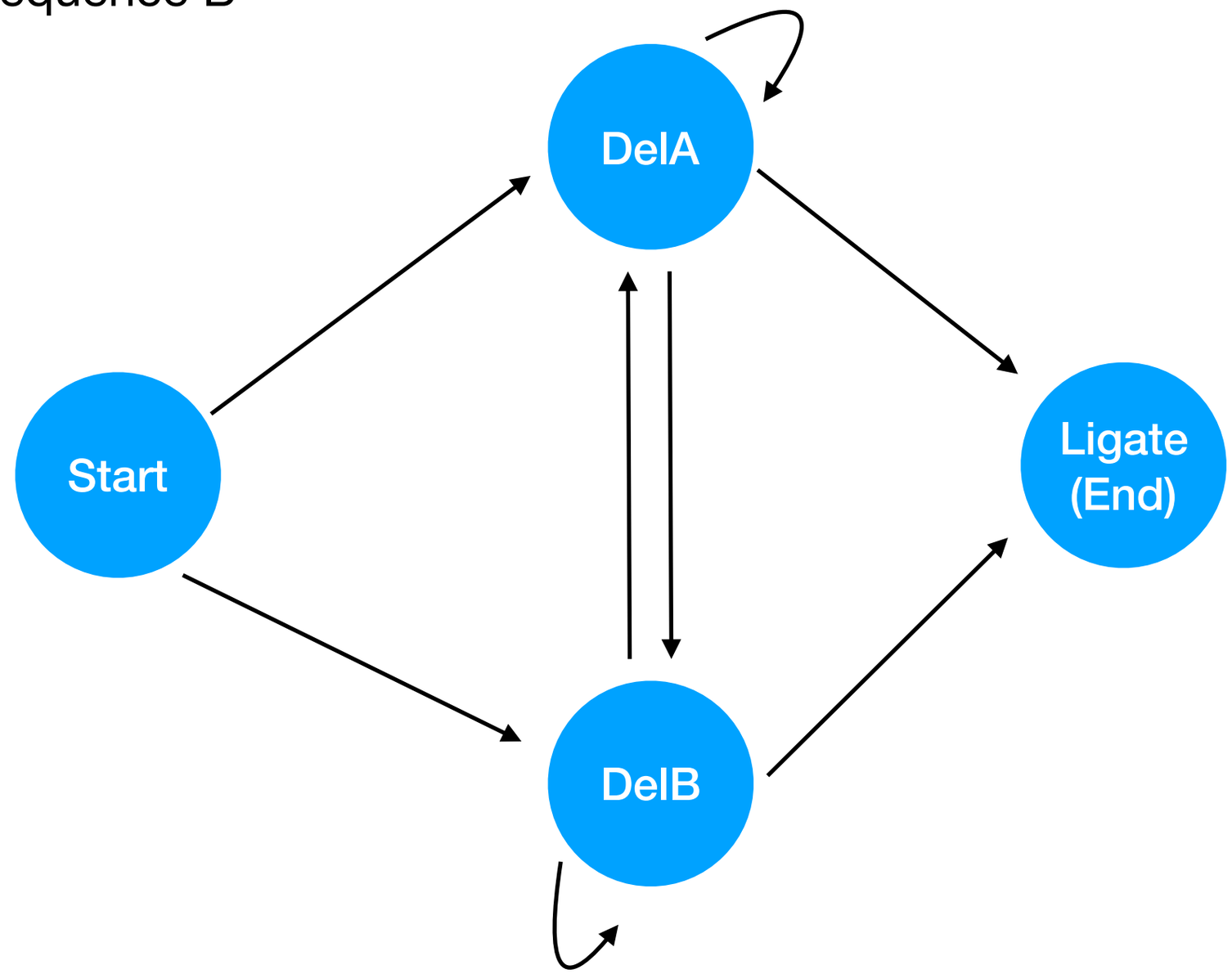
For example,



To move between states, some actions are available.
These actions are respectively: delA, delB, End

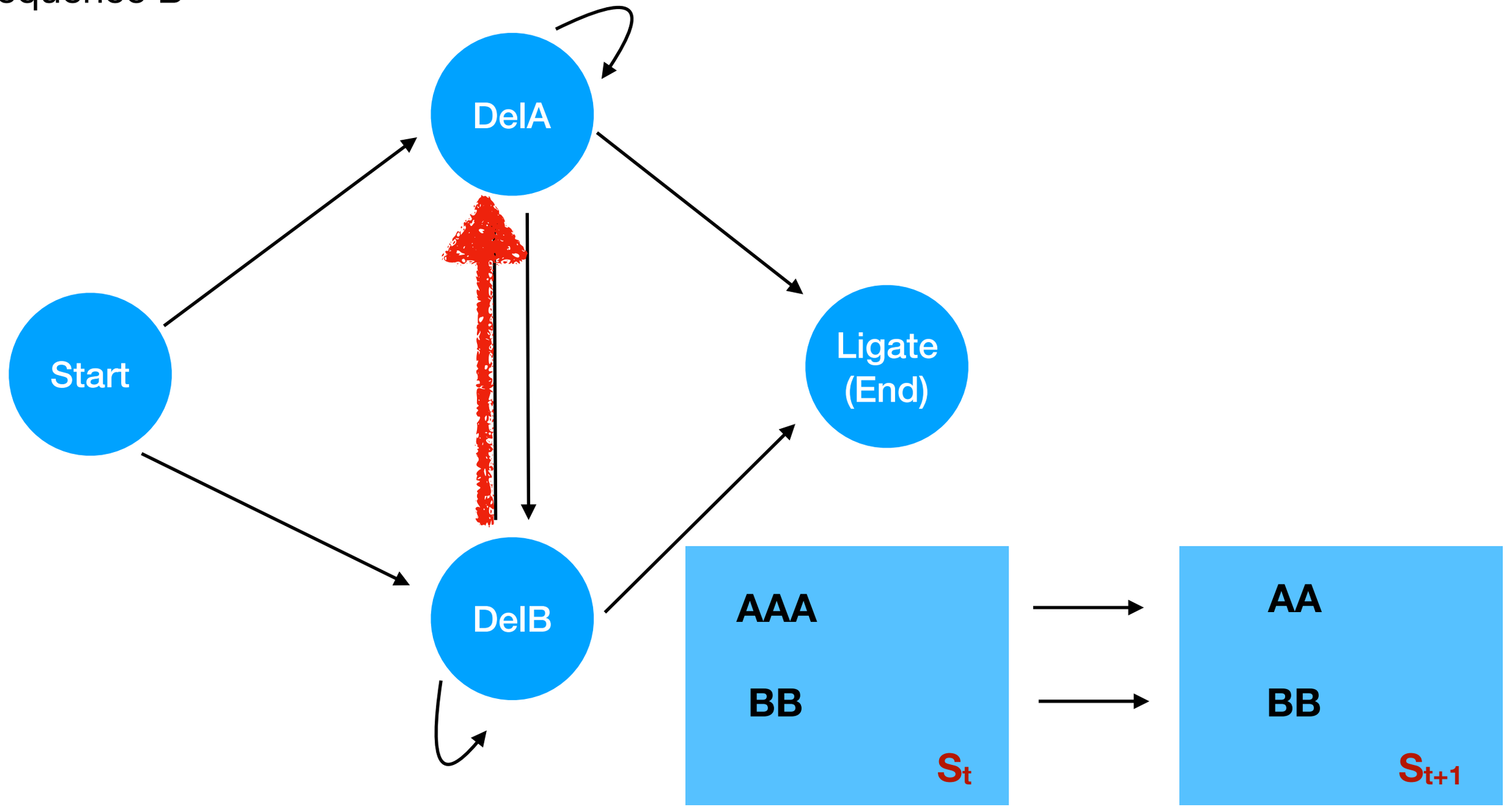
For the first step, we will disregard TdT activity and N insertions in the junction and simply concentrate ourselves on modelling trimming and ligation

Each chain of events starts at Start and ends at End.
DelA node deletes one nucleotide from the end of sequence A, DelB does the same for sequence B

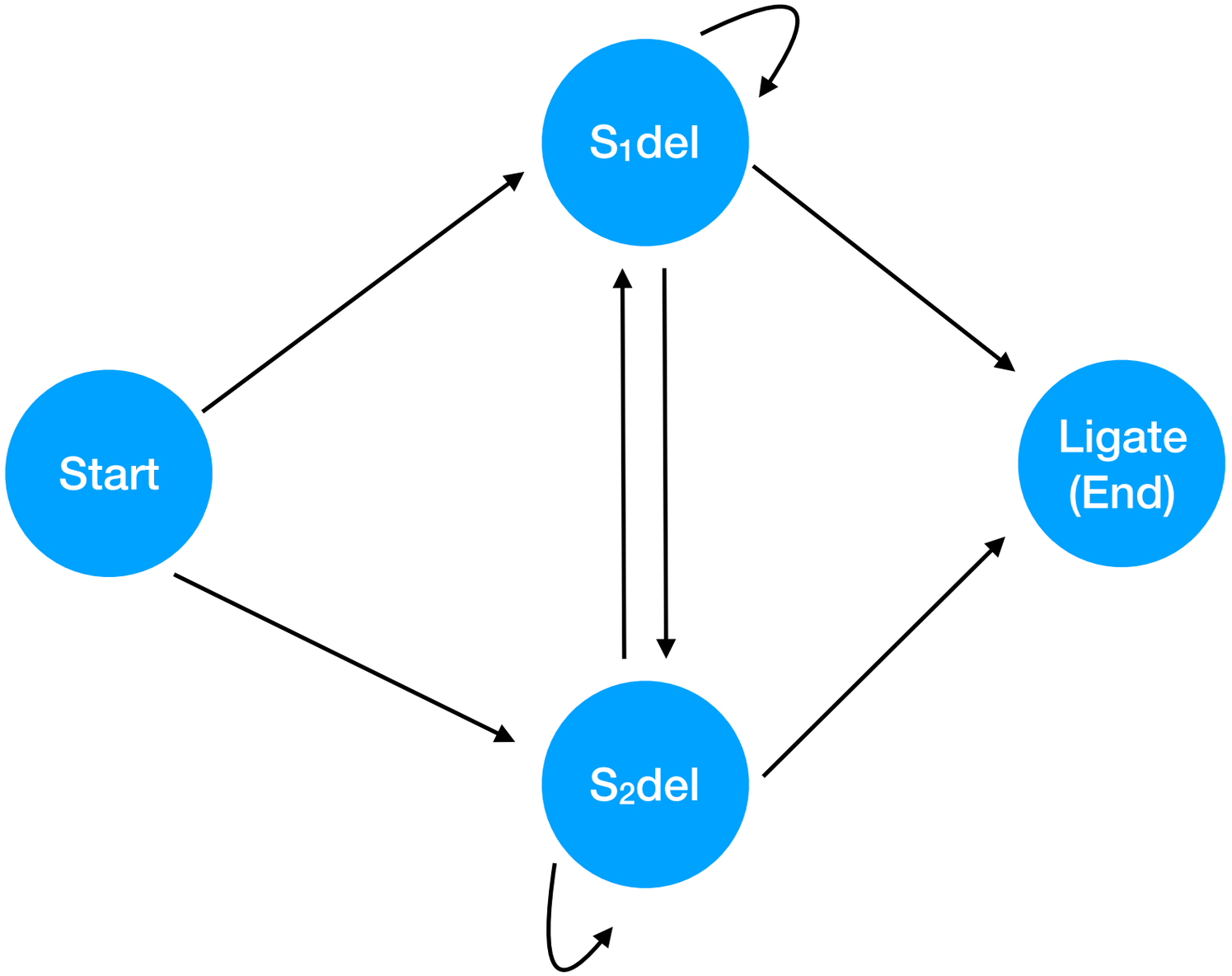


At any point in time, we transition from state S_t to S_{t+1} by visiting one of the nodes
What makes it even easier is we cannot ligate (END) until we get the correct subsequence for ligation

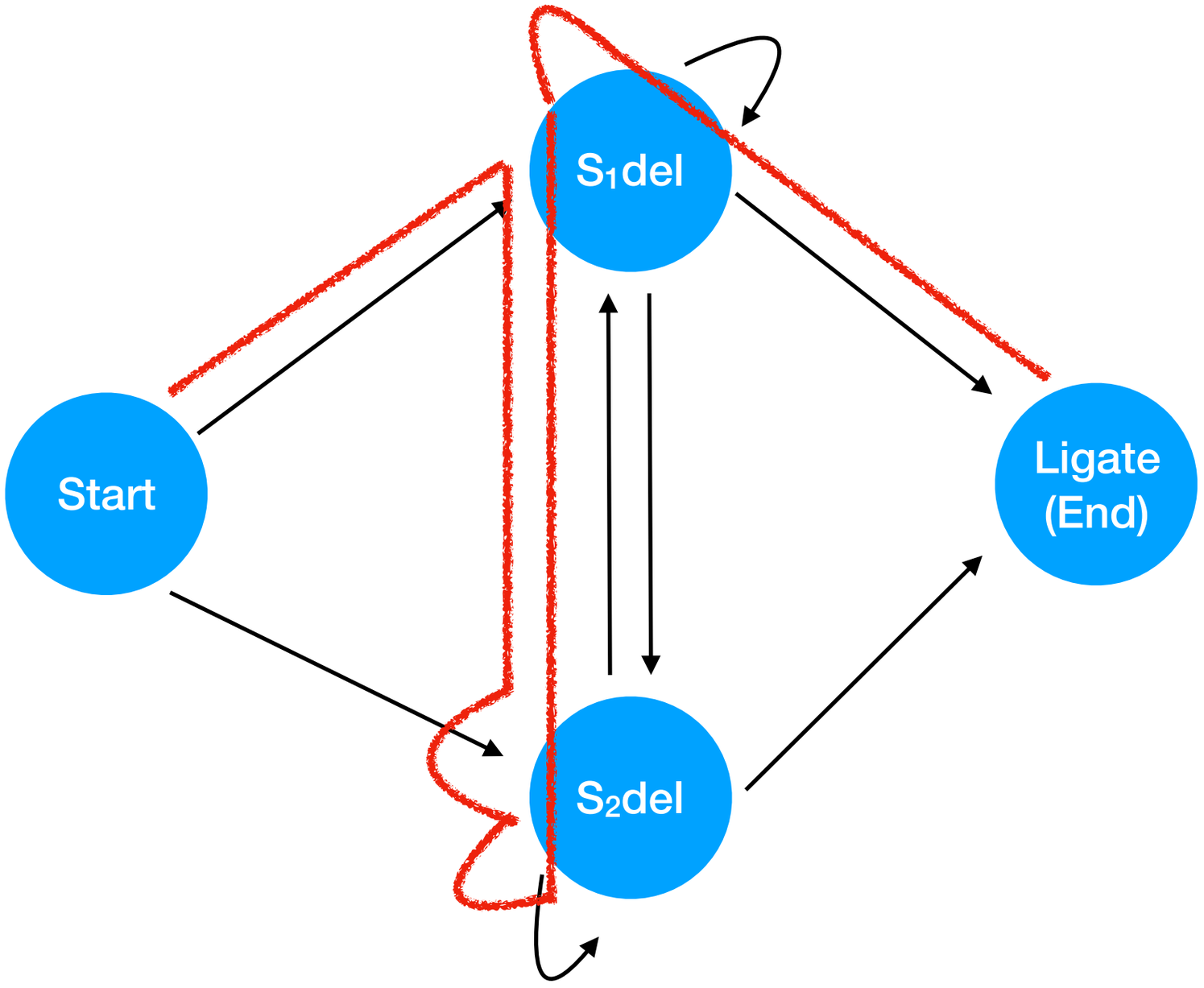
Each chain of events starts at Start and ends at End.
DelA node deletes one nucleotide from the end of sequence A, DelB does the same for sequence B



In this example, to transition from state S_t to state S_{t+1}, the agent visits the node DelA and trims off one nucleotide from sequence A

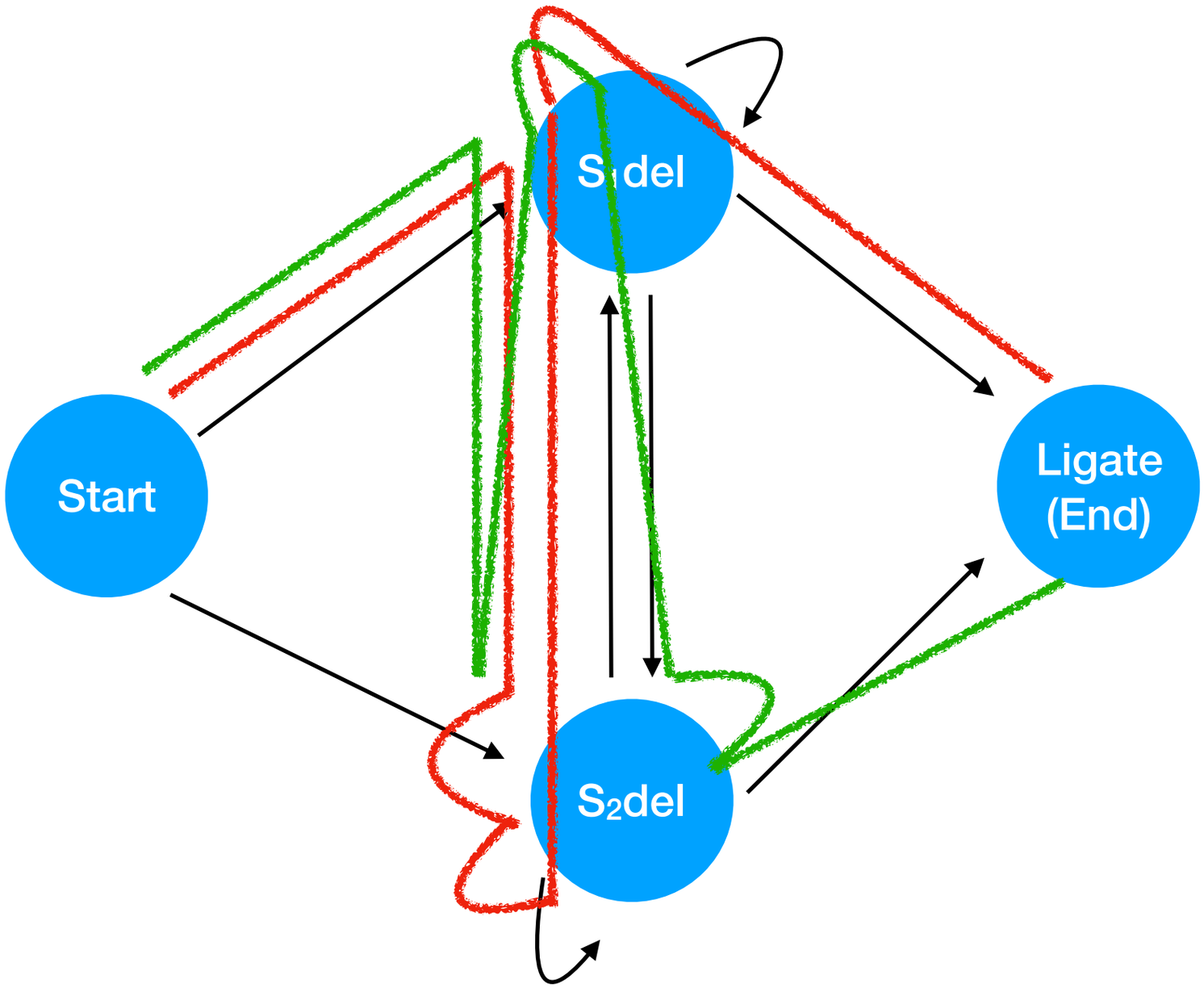


Since we know the subsequences and we know the total number of times we need to visit each node before ligation - the only thing that varies is the order of steps, based on our custom microhomology function M



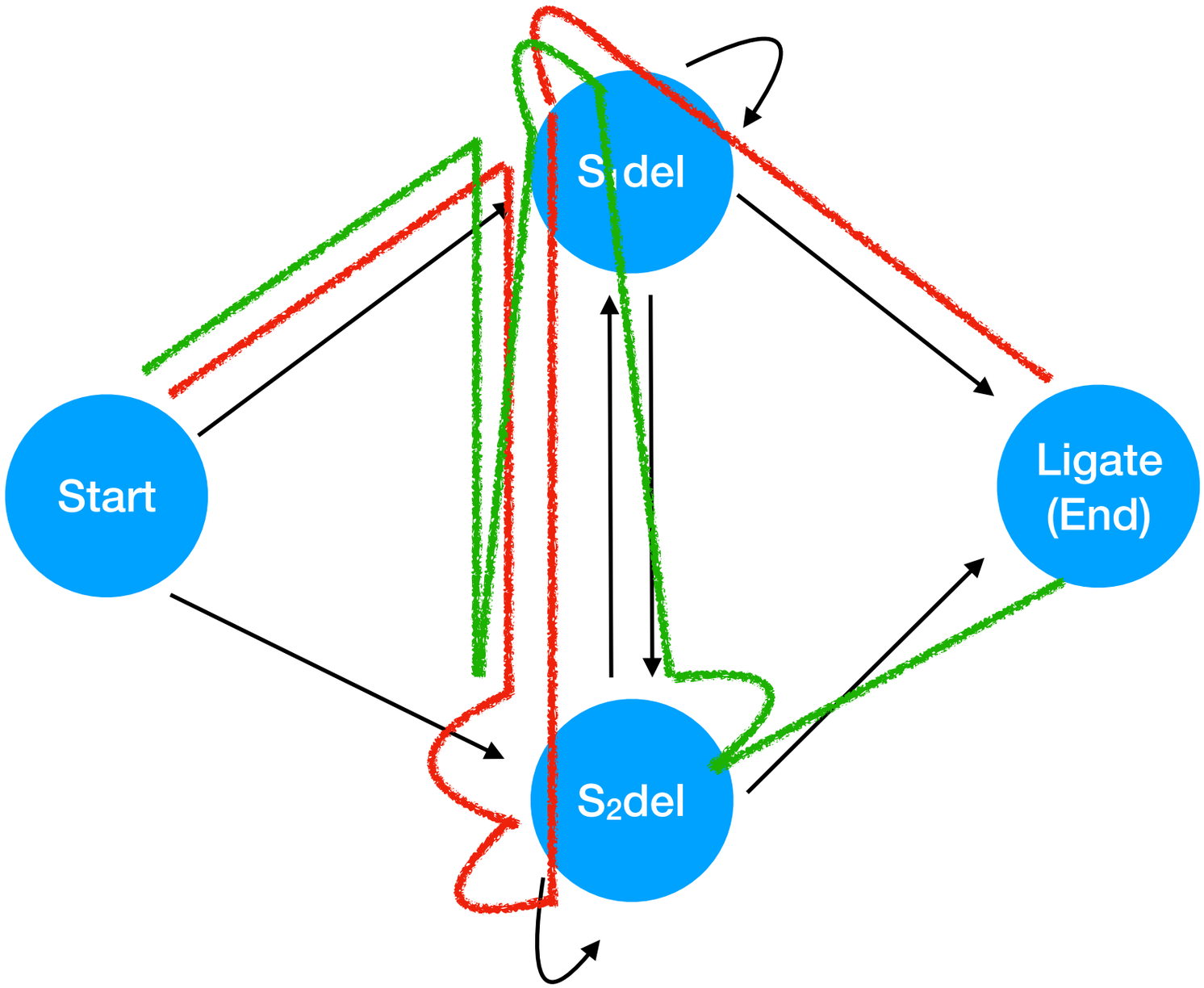
We can therefore create exhaustively all possible scenarios in visiting the nodes.

Start-S1del-S2del-S2del-S2del-S1del-S1del-Ligate



We can therefore create exhaustively all possible scenarios in visiting the nodes.

Start-S1del-S2del-S2del-S2del-S1del-S1del-Ligate
Start-S1del-S2del-S1del-S1del-S2del-S2del-Ligate
etc …



We can therefore create exhaustively all possible scenarios in visiting the nodes.

Start-S1del-S2del-S2del-S2del-S1del-S1del-Ligate
Start-S1del-S2del-S1del-S1del-S2del-S2del-Ligate
etc …
But then each scenario will have a score based on the steps!

We can therefore create exhaustively all possible scenarios in visiting the nodes.

We use a custom microhomology score function M(A,B) that calculates the microhomology between the trimmed off end and the kept end for both orientations

Start-S1del-S2del-S2del-S2del-S1del-S1del-Ligate **Score: 15**

Start-S1del-S2del-S1del-S1del-S2del-S2del-Ligate **Score: 21**

Then, we can rank the scenarios in order of highest score for a given M function
We can test multiple M functions.

Then, we can compare scenarios and their scores to sequence abundance in individuals, to see if the M function driving these scenarios has a bigger likelihood in the sequence distributions