Non-deterministic Turing machines in Prolog

A non-deterministic Turing machine (nTm) $M$ is specified by four finite lists, move-right, move-left, write-list, halt-list where

- move-right is a list of triples $[Q_0, X, Q]$ saying:
  from state $Q_0$, reading symbol $X$, move one square right, and go to state $Q$

- move-left is a list of triples $[Q_0, X, Q]$ saying:
  from state $Q_0$, reading symbol $X$, move one square left, and go to state $Q$

- write-list is a list of quadruples $[Q_0, X, Y, Q]$ saying:
  from state $Q_0$, reading symbol $X$, write symbol $Y$ and go to state $Q$ (keeping the head still)

and

- halt-list is a list of pairs $[Q_0, X]$ saying:
  from state $Q_0$, reading symbol $X$, halt.

Let us assume that every nTm $M$ has initial state $q_0$, and let us write $b-k$ for blank (a symbol indicating an empty tape cell). Let us agree that $M$ falls into a loop from which it can never halt, if it should ever be at a state $Q_0$ reading symbol $X$ such that none of its four lists specifies a next step.

Your task is to define a predicate $nTm(+move-right,+move-left,+write-list,+halt-list,+input,?output)$ such that the nTm specified by move-right, move-left, write-list and halt-list may halt with tape content $output$, given tape content $input$ (and initial state $q_0$ with the head at the leftmost symbol of $input$). Let us assume without loss of generality that $input$ is a non-empty list, using the list $[b-k]$, if necessary, to represent the empty string. As for the output string, on the other hand, let us arrange that it never begins or ends with a blank (so that for example the list $[b-k]$ is rewritten as $[]$, and $[b-k,a,b-k,b,b-k]$ as $[a,b-k,b]$).

Hint

Take “snapshots” $[L,R,Q]$ of a run of a nTm, where

- $L$ lists the non-blank tape contents to the left of the nTm head (in reverse)
- $R$ lists the non-blank tape contents to the right of the nTm head, including the currently scanned cell (the head of $R$)
- $Q$ is the current nTm state.

Search through a graph with nodes $[L,R,Q]$, the children of which arise from a single nTm step.

A query

$nTm(MR,ML,WL,HL,Input,Output)$

sets the start node (the root of the computation tree) to $[[],Input,q0]$. 
Sample runs

?- nTm([],[],[],[[q0,X]],[b-k,i,n,b-k,p,u,t,b-k,b-k],Out).
X = b-k,
Out = [i,n,b-k,p,u,t] ? ;
no

?- nTm([[q1,1,q2],[q1,0,q2],[q1,b-k,q2]],[],
[[q0,0,1,q1],[q2,0,b-k,q1]],
[[q1,b-k]],
[0,0,0,0,0],
Out).
Out = [1,b-k,0,0,0] ? ;
Out = [1,b-k,b-k,0,0] ? ;
Out = [1,b-k,b-k,b-k,0] ? ;
Out = [1] ? ;
no

?- nTm([[mr1,h,we],[mr1,e,wl],[mr1,l,wp],[mr1,p,wbk],
[mr1,l,wo],[mr,o,wo],[mp,o,wp]],
[[q0,0,lbk],[lbk,b-k,lbk]],
[[q0,0,h,mr1],[we,1,e,mr1],[wl,0,l,mr1],[wp,1,p,mr1],
[q0,0,l,mr],[wo,1,o,mr],[wo,0,o,mp]],
[[hbk,b-k]],
[0,1,0,1],
Output).
Output = [h,e,l,p] ? ;
Output = [l,o,o,p] ? ;
<loops>