state F at time $t(n) = 2n \quad 1$ *if the initiator is located at the left endpoint of the array or time* $t(n) = 3n + 1$ *if the initiator is located at the right endpoint.*

Finally, in Section 4.4 we present a 7-state non-minimal-time solution to the generalized problem mentioned above, where the initiator can be anywhere in the array. The transition function for the automaton may be found in Table 12. This automaton has the fewest states of any known solution to this problem and requires 2 fewer states than our minimal-time automaton. We also have a proof of correctness for the 7-state solution which shows that a one-dimensional array of n automata with initiator located in position k will synchronize in $2n \quad 2 + k$ time steps for any $n \quad \mathcal{N} \; n \quad 2$. The details of the proof are omitted but may be found elsewhere [14].

Few researchers in this area have provided correctness proofs for their automata. Indeed, as far as we know, prior to this work only Balzer [1] and Mazoyer [6] have published proofs of correctness.

Our work provides additional evidence that Mazoyer s conjecture does not hold by giving non-minimal-time solutions to both the original and generalized versions of the ¯ring synchronization problem which require fewer states than the best known minimal-time solutions. Indeed optimal non-minimal-time solutions may use even fewer states than our constructions, as our automata are based on minimal-time solutions to restricted versions of the problem.

1.4. Summary of results

Results for both the original ¯ring synchronization problem and the generalized ¯ring synchronization problem are summarized in Tables 1 and 2. Table 1 gives the minimal-time solutions and Table 2 lists non-minimal-time results.

2. The ¯ring synchronization problem

The ¯ring synchronization problem, sometimes also called the ¯ring squad problem, is a classical problem of synchronization. Consider a one-dimensional array of n ¯nite automata in which all automata are identical except the ones on either end of the array. The end machines di er from the interior machines in that each is aware that it is at the end of the array. It is possible to use a single transition function for all machines if we allow an end marker next to each end machine and use this marker in de¯ning the transition function. This end marker does not count as a state in the solution.

The automata in the array work synchronously, and the state of an automaton at time t only depends on its and its neighbors states at time $t \quad 1$. At time step 0 of the computation all automata are in a special quiescent state except for one automaton that is designated as the initiator for the synchronization. In the original ¯ring synchronization problem this machine must be located at the end of the array. The problem is to de¯ne the set of states and transition rules for the automata so that all machines enter a special ¯re state simultaneously and for the ¯rst time at some time $t(n)$.

The transition function for each automaton can be given as a set of 4-tuples. The 4-tuple (X,Y,Z,W) represents the rule that an automaton currently in state Y, with left neighbor in state X and right neighbor in state Z will enter state W at the next time step. We will denote this by XYZ W. By de¯nition automata solving the ¯ring synchronization