A white paper with black text

Description automatically generated

Let's prove the statements using induction:

1. If v = v', where v is the value produced by the minimax algorithm and v' is the result of running Alpha-Beta Pruning with some initial values of α and β:
   1. Base Case: If s is a terminal state, both minimax and Alpha-Beta Pruning produce the same value, which is v = v'.
   2. Inductive Step: Assume the claim holds for all children of s. In the minimax algorithm, v represents the best possible outcome for the current player. Similarly, in Alpha-Beta Pruning, v' represents the best possible outcome within the bounds of α and β. Since the bounds α and β are tightened as we traverse the tree, Alpha-Beta Pruning will not exceed the minimax value. Therefore, v = v'.
2. If v < v':
   1. This implies that the minimax value is less than the value produced by Alpha-Beta Pruning. Since Alpha-Beta Pruning only prunes branches where the value is worse than β, the value v' is a conservative estimate of the true minimax value.
3. If v > v':
   1. This implies that the minimax value is greater than the value produced by Alpha-Beta Pruning. Similarly, since Alpha-Beta Pruning only keeps values between α and β, the value v' is a conservative estimate of the true minimax value.

Therefore, the hypothesis holds true for all cases: if the true minimax value is within the range defined by the initial values of α and β, then Alpha-Beta Pruning returns the correct value. If the true minimax value lies outside this range, Alpha-Beta Pruning may return a different value, but it is bounded by the same constraints as the true minimax value. Therefore, Alpha-Beta Pruning will be correct with initial values of (-1, +1) for (α, β).A white background with black text

Description automatically generated

To prove that any n-ary constraint can be converted into a set of binary constraints, let's consider a constraint involving n variables: (x1, x2, ..., xn)

1. Create Synthetic Variables:
   1. Introduce a synthetic variable, Y, with a domain that is the Cartesian product of the domains of (X1, X2, ..., Xn). The domain of Y would consist of tuples (x1, x2, ..., xn) where xi belongs to the domain of Xi for i = 1 to n.
2. Replace Constraint w/ Binary Constraints:
   1. Replace the original n-ary constraint with a set of binary constraints.
   2. Each binary constraint will relate Y with one of the original variables, Xi, for i = 1 to n.
   3. For each tuple d = (x1, x2, ..., xn) in the domain of Y, create a binary constraint between Y and Xi stating that Xi = xi.

These binary constraints will effectively enforce the original n-ary constraint.

By introducing synthetic variables and transforming the n-ary constraint into a set of binary constraints, we've shown that any n-ary constraint can indeed be converted into binary constraints. Therefore, all Constraint Satisfaction Problems (CSPs) can be converted into Binary CSPs by representing each n-ary constraint in terms of binary constraints.