# Left Message Passing Strategy

Note：将一个X加入到一个中间节点的ES结构中（ZR）分两种情况：

1. Transfer
2. Infeasible，此时会替换最小权值，参照StaticOIS中的流程

由函数TreeNode:: insertXintoESinNode实现

左侧消息传递策略（待证明）

1. 左侧插入成功
   1. 直接加入到P.ZL与P.Z集合中
2. 左侧插入失败
   1. 左侧Transfer
      1. \_bZ在P.Z中（是否等价于在P.ZL中？）
         1. 使用EE结构求解\_aZ在P.ZL中的Replaceable Set R
         2. 求出R中end最大的元素maxEndInR
         3. 从P.Z和P.\_ZL中删除maxEndInR
         4. maxEndInR加入到P的ES结构中（即ZR）
         5. 根据结果更新Message
      2. \_bZ不在P.Z中
         1. 使用EE结构求解\_aZ在P.ZL中的Replaceable Set R（为何可用EE结构求解R？待严格证明）
         2. 求出R中end最大的元素maxEndInR
         3. 若maxEndInR.e<P.rightChild.Y.s（由L中的transfer变为P中左侧的infeasible）
            1. 求出R中w最小的元素，从P.Z与P.ZL中删除，再将该元素进到I中
            2. 更新Message
         4. 若maxEndInR.e>P.rightChild.Y.s（依然transfer）
            1. 在P.Z与P.ZL中删除maxEndInR
            2. maxEndInR加入到P的ES结构中（即ZR）
            3. 更新Message
   2. 左侧Infeasible
      1. \_bZ在P.Z中（是否等价于在P.ZL中？）
         1. 在P.Z与P.ZL中删除\_bZ
         2. Message不变
      2. \_bZ不在P.Z中
         1. 使用EE结构求解\_aZ在P.ZL中的Replaceable Set R（为何可用EE结构求解R？）
         2. 求出R中end最大的元素maxEndInR
         3. 若maxEndInR.e<P.rightChild.Y.s
            1. 求出R中w最小的元素，从P.Z与P.ZL中删除，再将该元素进到I中
            2. 更新Message
         4. 若maxEndInR.e>P.rightChild.Y.s（由L中的infeasible变为P中的transfer）
            1. 在P.Z与P.ZL中删除maxEndInR
            2. maxEndInR加入到P的ES结构中（即ZR）
            3. 更新Message

后续工作：

1. 整合右侧消息传递内容（整合以后左侧的消息传递策略可能会产生错误，不确定）。
2. 分裂节点时维护相应ZIT集合的优化算法。目前算法时当发生分裂时，清空L与P中的Z、I、T，然后在L中全部重新插入，最后进行一次左侧消息传递。即目前分裂中的代码与左侧消息传递中的代码时一致的。
3. 优化传递算法，消除重复部分。