**第一篇文章：**

这篇文章引入零一变量x对路径图进行编码，引入整数规划模型，通过添加限制，实现规划。

在基于路径图的编码中，为了防止智能体重复进入已经访问过的节点，借鉴了旅行商问题引入了限制，并且根据QCOP方法，防止产生子循环（这里引入限制：ui −uj +1≤(|V|−3)(1−xvi,vj). u表示节点被访问的顺序。这一不等式可以保证在除去初始以及目标节点后，若节点i和节点j都在P路径中，则可以保证ui −uj +1≤0，ui < uj，即先对i访问过后再访问j，如果可以保证节点i和j是相邻两个被访问的节点，则可以满足图中仅有小于等于一个循环路径（对于循环路径来说无法恒满足先访问节点i后立即访问节点j），但是对于xvi,vj的定义和引用文献中的定义我感觉有些出入，这里定义：x=1当且仅当节点i和j在路径中，但是引用文献（Correlated Orienteering Problem and it Application to Persistent Monitoring Tasks, Jingjin Yu, Mac Schwager, Daniela Rus）中定义：x=1当且仅当节点i之后立即访问节点j所以我认为在本篇文章中该限制可以确保两个在路径p中的节点会相继访问，顺序必定是先i再j，对于除初始节点出发的路径外的子循环来说，无法满足ui恒小于uj的条件，所以可以保证不存在子循环。

另一种路径编码方式是基于时间的，在有限时间T内对V进行T+1个复制，引入零一变量x指示两相邻时间点所访问的节点i和j是否在路径P中。另外还基于时间编码方式对多智能体路径规划的时间进行优化模拟，引入两种创新的方法：相邻两节点中建立通道以及建立可达圆，以此框定出可行域。此外，这篇文章对多智能体路径规划中障碍排除问题也进行了叙述，通过构建能反应障碍物覆盖范围的图来减小最小障碍物移动问题的状态空间，通过限制变量，进行整数规划，从而最小化移动障碍物的数量，并且对于多智能体系统来说，这种方法能够降低计算时间，但是对于单一智能体来说则并没有改善计算时间，但是我对于这个方法并不是很理解，是否是通过对限制的处理来框定变量x的范围，然后通过对于常数L的选择，来最小化障碍物移动的数量？最后对于收集问题RCP，基于对路径进行时间编码，以此适用于整数规划模型，使得采集量最大化。但是对于这个问题，Une image contenant objet, montre

Description générée automatiquement，这一从文献中给出的收集量最大化的公式中，我并不能理解其如何实现最大化，xi(xj-xi)必定恒小于等于0，那么对于vi领域中的vj，所采集的数量则是一个负数，rixi加上一个负数，那么采集量应该会降低，我查阅了引用的文献，发现引用的文献中给出的公式是将xj-xi换为xi-xj：

Une image contenant objet

Description générée automatiquement

这个式子我觉得我可以理解，式子右半部分的第二项在节点j不在路径内的情况下，可以通过在节点i是收集节点j的部分收集量，即rjwij，接下来就可以根据文章所得出的结论来最大化收集量。

对于运筹学中的整数规划来说，我认为最关键的还是找出限制条件，找出限制条件，应用IP模型，可以进行规划。

**对第二篇文章：**

基于SMT（满足性模理论）的路径规划，我认为这篇文章和第一篇基于整数规划的路径优化问题有相似之处。这篇文章提出了SAT-TSP算法，上一篇文章中也应用到了TSP旅行商问题，和上一篇文章一样，作者将图进行零一布雷尔变量编码，以表示节点的访问情况。这篇文章所应用的DPLL算法我并没有完全看懂，但是从表达形式上来看，似乎比上一篇文章以将SAT编码后以求和的方式给出限制的方法会简洁一些，但是逻辑要求比较高，以and和or的方式表示出限制，将限制表示为clause的组合。

这篇文章中应用到了多个求解器，并且直接将这些求解器应用到了算法中，比如SAT布雷尔可满足性求解器，应用该求解器来解出系统的限制，找出满足限制的候选选择，后应用TSP旅行商问题求解器求解出最优解。

文章所提出的BRUTE算法求解SAT-TSP实例，通过先求解出SAT实例，然后再求解TSP实例，是暴力求取最优解的方法，但是思维难度较低，首先使用SAT求解器找出满足限制的可行顶点组V’，然后调用TSP求解器求解出满足限制的最小cost路径p’，这样的算法虽然逻辑简单，但是计算量较大，因此在BRUTE算法基础上拓展出了cbTSP理论，基于部分解来缩减计算空间，使用DPLL算法耦合SAT和TSP求解器，通过部分解M来缩减图G，得到TSP单调子图，之后构建SAT-TSP问题并求解，通过判断cbTSP理论的满足性来找出满足限制的解，通过这些解，使用binary search来找出满足最小cost的解。

这篇文章出现了比较多的新理论，比如np-complet（NP完全）等理论，所以理解起来存在着一些困难。

**第三篇文章：**

基于采样的LTL多机器人路径规划问题。这篇文章旨在解决在机器人数量较多的情况下基于Graph的算法复杂度较高的问题，基于时序逻辑，而在第二篇文章里也提出了应用LTL（全局线性时间逻辑）解决路径规划问题。这篇文章的作者提出了基于采样的方法，由于现有的LTL方法是基于Graph的，为了降低复杂度，作者提出了基于抽样的算法，通过建立有向树来逼近同步积自动机的状态空间和状态转移。同步积自动机这一概念我是第一次接触，包括形式语言等。在基于有向树的算法下，通过跟踪父节点-叶节点最后到根结点来构造出运动路径的方法复杂度较低。

文章引入了自动机、形式语言这些概念，构造出对于每个机器人的weighted Transition System（wTS）来描述运动，并且给出了成本函数，目标是将成本函数最小化。

算法将运动规划分解为prefix部分和suffix部分，我称之为前缀部分和后缀部分，前缀部分的算法目的是找出从初始状态到目标状态的路径，只运行一次，而后缀部分则持续运行，目的是找出最终状态附近的循环，循环始于最终状态，终于最终状态，以此逼近product automaton。

构造前缀部分的算法起始于起始状态，通过sample（采样）、Extend（拓展）、Rewire（重装），Optimize（优化）最终找出从初始状态到最终状态的路径，并且通过在Extend部分中添加的argmin保证找出cost最小解，并且对每一个最终状态都规划出从初始状态出发的路径。

构造后缀部分则是从构造前缀部分所得到的最终状态出发，算法和构造前缀部分的算法类似，目的是在最终状态附近存在循环，始于最终状态，终于最终状态，若不存在，则按照前缀部分相似的构造方法构造出一个路径，以满足条件，并且找出满足cost最小化的循环路径。

由于我不理解形式语言、自动机的概念，所以我并不理解为何要在构造有向树时需要在每个最终状态构造一个最小循环。我在FPGA自学里了解到的状态机的概念是基于状态转移的，与这里类似，但是似乎并不需要满足每个最终状态周围都存在一个始于最终状态终于最终状态的循环，我能理解其算法，但是并不理解其存在的原因，还希望老师能够给予解答。