

订阅DeepL Pro以编辑此演示文稿。  
访问[www.DeepL.com/pro](https://www.deepl.com/pro?cta=edit-document)，了解更多信息。

他们会把你关在这里 哦，不不不，对不起。 不不不不不 这是LP二，LP三和四。 因为我们还得把 X 1 分支出来 所以，这个分支将X 1小于等于1X1。 我只数了2 一个？ 没错 对不起，伙计们。 这是LP一个我们可以分支。 但你会是不同的。 是的，对不起 Yes. 抱歉 所以，在这种情况下，让我们在这里提醒你 LP零V分支。 哪一个？ 这里 对 - 不对 不对 还有这个 这一个比一个少。 一个 一个 是的 是的 Yes. 是的 是的 Yes. 好吧 - Yes. 好的 现在我们开始记录 Okay. 现在，我们正在跟踪。 好的 这是放松线性Luxation 加上约束X 2 Greatrillioneporting 2 LP 3 Linearisation plus this constraint plus this constraint okay. 对 很显然，我们可以 我们可以确定 让我们假设五和六 从这里在这种情况下将是。 X一小于等于二或大于等于。 好极了 好的，很好 让我们来看看结果。 LP 3 LP 3是正确的。 X 2大于2 2 好极了 X 1 小于一。 完美解决 X11X true. 分数。 再来一次 是的，我知道这是感觉对象函数 12 点。 当我们解决LP为。 不可行 好吧LP为不可行的约束。 哦 对不起 可行性约束的可行性。 好吧，伙计们，我们不能进一步分支。 显然，我们不能再分支了 因为不可行，就没有解可以分支了 好吧，在不可行之前帮帮忙。 好极了 所以，我不需要关心这个问题。 但LP又来了 这里有分数解 从LP得到LP4 现在你有了 LP. 五是修正 哦，不，不，不，不。 耶稣基督。 这是在这里。 癸 是的，我怎么样？ How am I? 上帝，是的。 是的 好极了 Great. 好了，第五张唱片 Okay. LP五。 你是这个问题加。 算算看 对，没错 对 - 对 对 - 对 对 - 耶！ 对 好了，LP三，LP四 让我们确保LP五 LP五。 好的 Perfect x two. 小于2 LP 6X2 大于等于三 好极了 只要解决 LP 五解 X11X22。 对 整数最优值 11。 这意味着这是最优解吗？ 不是，但如果它是整数，它就是整数。 现在事情开始变得有点有趣了，因为这是第一个整数。 当出现这种情况时，解法就像是法语中整数编程的一个事件。 关于。 因此，这个解决方案现在有了一个不同的名字。 它是一个现任解决方案。 现任方案是目前最好的方案 这并不意味着它是最优解 最终解决方案是最优的。 但我们可以肯定的是，你们不在这里。 我们还在这里 好吧，我们没有解决这个问题。 所以在求解方法上，LP5是我们得到的最好的。 求解结果为1 2，最优值为11。 现在，我们终于更新了下限。 现在我们知道，在最坏的情况下，我们的最优解将是 11。 好的，但结果是我们解出了 LP 6。 好的，所以这个解 k 是整数。 所以这里就没有分支了。 为什么？ 因为一切都是整数。 没有办法继续分支 LP 五。 好极了 所以它的边界是完整性 或者在这种情况下，可行性二。 在这种情况下，LP六。 我们解决了 再来 我们解决了 这是一个X10和X到3的最优值12。 这重要吗？ 没有，所以下限现在更新为 12。 因为你不知道它是否最优。 你怎么看？ 你觉得如何？ 12. 你觉得我们能得到比12更好的解吗？ 好的 你还记得12点线性Luxation的值吗？ 75. 你知道75 我们能得到的最好值 这是最好的，你可以得到的。 是12 但这并不意味着最终你可以分支剩余的节点，比如说，你可以分支LP一，以确保你能在家里做到这一点。 你可以分支 LP 一，但肯定找不到比 12 更好的解。 所以，最终你会有多种选择。 所以这些选项应该是一样的。 x 一和 x 二最终会有不同的值，但最大值是 12，我们不可能找到比 12 更好的解。 所以我不知道这样做对不对。 对，我想是的。 所以，我们开始像这样的路线解决方案12.75注我从来没有探索这一个好吧，我做了下一个。 基本上我不需要，因为我知道基本上最好的解决方案是12，我会是10或12的解决方案，因为我是这样的，好吧，这就像这个方向。 但很明显，我可以在这个LP之前分叉。 但我们肯定知道这里最好的解是11 因为说到底 这是对原始问题的放松 没错 这就是我们常说的 "内向三分支"。 很明显，它适用于非常小的问题，两个三个变量，甚至不是三个变量，两个变量加上几个约束条件。 因为很明显，你可以看到在这个案例中，我们最终解决了多少个 LP。 一个 23456. 非常简单 两个变量。 三个约束 这样说吧，一个约束条件和两个约束条件，基本上都是初始决策变量的约束条件。 即使是这个非常简单的简化问题。 最后，我们还是得到了六个 LP。 试想一下，我们有一个真正的 LP，其中有成百上千亿个决策变量。 因此，当他们少讲课时，就会明白，正是少讲课才显示出差距在迭代改进。 因此，差距基本上是指下限和上限之间的差距。 在这种情况下，很明显，我们设法找到了最优解，因为它是最后一个被利用的节点。 我们杀死了所有节点，或约束了其他节点，或修剪了其他节点。 好的。 虽然方法各有不同，但归根结底，这就是智能枚举的主旨所在。 因为如果不智能，我就需要枚举所有潜在的。 解决方案 这意味着，要对所有潜在节点进行分支。 而在这种情况下，比如说，我不需要分支这个节点。 因此，这才是枚举的智能部分，而不是穷举，因为确切地说，界限程序最终是因为我知道或我就是下界和上界，我知道不可能从给定节点中找到好的解决方案，我可以修剪或杀死或终止给定的探索，在这种情况下会发生这种情况，因为在这种情况下显而易见。 边界 11.5. 考虑到我们曾承诺过更好的下限。 好吧。 尽管如此 是的。 如果我们有 A 我的而不是我们。 1130 20. 所以，对不起，你得到了11。 是11. 0 我们得到了 13. 虽然我们还是得到了13. 我知道这是关于12点和一个例子。 是。 好了一边和一个较低的整数金额 在另一侧时的品牌。 那么，在这种情况下，例如，如果你正在采取更好的东西，例如，你应该探索这些节点，基本上给一个更好的或承诺一个更好的解决方案。 这是一种更好的放松，所以最终你可以从这个节点得到更好的解决方案。 显然，13 这种情况是不可能出现的，因为我们知道最佳值解应该是 12。 但是，如果是 11.8 或类似的值，你就会选择这个节点，开始探索这个节点。 最终，这可能是可行的。 如果是这样的话，你就需要探索另一个节点，然后得到一个最优解。 因此，我们知道可能的最高数额是 12.75。 因为这是一个最大化问题，而 12 是最近的。 这是最近的。 你能做的就是用 12 作为界限，但你不知道最优解是否正好是 12。 这就是问题所在。 四舍五入法可以给我们一些关于最优解的直觉，但你并不确定。 最终 这是一个非常简单的问题，但不是一个全年都能解决的问题。 还记得舍入练习吗？ 我们用不同的方法求解，但都没有接近整数最优解。 所以，一般来说，当你掌握了更多关于问题结构的信息时，你可以使用四舍五入程序，在下界和上界之间开始一个周期。 本例中我们没有这么做，但可以把它作为上限。 但在本例中，上限 是 12。 但你不知道这对你的问题是否可行。 你不知道因为有限制条件，你最终能得到 12。 你不可能得到 12，你可以得到 11。 这就是问题所在。 但这只是一个约束条件。 请举手 先看分支和边界的脑子。 让我们先正式确定一些术语。 分支指的是解空间的分割过程。 我们有一个 LP，它有一个分数解。 我们选择分数解作为分支的起点。 然后我们开始分割解空间。 分支过程可以看作是对可行区域进行的一次又一次的细分，其中每个子集都是一个给定的分区，代表了一个问题。 为什么是分割呢？ 因为，正如你在这里所注意到的，分区是因为没有一个解决方案能同时满足这两个条件。 同时添加的约束条件。 好了，这就是为什么这是一个分区。 解要么在这里，要么在那里。 在这种情况下，因为你看到这两个区间之间没有交集。 在这两个区间之间。 这就是为什么这是一个分区，而不是一般的分区。 分区的概念是分区的交集是空集。 什么都没有。 交集的分支过程也可以看作一棵树。 分支方法有三种，其中根节点代表线性Luxation。 好了，这里是原 IP 整数程序的 LP 松弛表述，每个节点代表一个潜在的子问题，其中的约束条件是指我们分割空间时，原约束条件加上其他约束条件。 一般来说，如何实现分支。 假设我们有一个给定的分式解，或者没有给定的分式解。 积分解 x I，其中 x I。 对不起，应该是 I q，其中 x I 介于 I 1 和 I 2 之间。 所以，基本上 I 一和 I 二是两个连续的非负整数。 举个例子 我们有 1.5，所以 1.5 我们有一个，你有它到两个连续的整数。 这就是我想要的。 我展示了连续的非负整数。 然后，在连续近似程序中加入小于或等于我想要的约束条件 x 1，或者大于或等于我想要的约束条件 x I，创建两个新的循环。 这就是一般形式。 什么？ 我们刚才对分支边界做了什么？ 分支过程具有缩小可行区域的作用。 什么是缩小？ 因为基本上我们有一个给定的 所以我们有一个给定的可行区域，当我们开始加入这些约束或这些边界缩小时，会发生什么呢？ 我们正在 包括边界和约束条件。 所以我们在缩小。 好的 缩小可行区域 但有趣的是，在这种情况下，当我们这样做的时候，我们正在摆脱什么整数或分数解。 没错 我们要去掉小数解。 这是主要的启示 显然，我们在给定每个分支的时候，并没有去掉任何整数解。 好吧，还有什么？ 还有什么？ 从进一步考虑的角度来看，当前的非积分解或 x I 的小数解，或者正在打印但为真的变量，都会保留所有的整数解。 好吧，显然是原问题。 最重要的是，这个过程被证明是收敛的，因为我们有一个有限的可行区域。 如果存在可行解，这个过程会在几次迭代后终止。 显然，这可能会非常非常耗时，但理论上，由于可行区域或最优整数解的角点数量有限，总有一天算法会终止。 就分支的边界而言。 分支边界。 分支边界指的是一种边界方案，基本上是用来消除某个给定节点的。 因此，我们基本上有三种类型。 别谈这个了。 我们有三次被隐形破坏的结合程序。 这说明，基本上我们的解法是不可行的，或者基本上是不可行的。 没办法。 我们要么剪枝，要么杀掉，要么因为不可行而终止这些节点。 LP 解都是整数。 在这种情况下，我们丢弃这些节点，并说这就是剪枝。 还记得第一个吗？ 第一个内部解 11 中没有任何分支解。 这可能是最优解。 好的 比如说，当我们知道这个节点不可能有更好的解时，因为我们已经有了更好的解12。 所以我们就不需要处理这个了。 不再处理了 没错 因为有边界 因为我们知道这个LP松弛，LP值是11.5。 最好的情况下，最优解是 11。 所以，考虑到我们已经有了 12，所以 11 也没什么可做的了。 所以在这种情况下，我们可以进行边界修剪。 好的，因为在这种情况下，我们发现了一个更好的边界，很明显，一个积分或整数解的边界更好。 从一些非常普遍的规则来看。 所以，当你有一个 LP 解，并且有多个 LP 解时，你要处理的是一个大规模的问题，多个分数解。 下一个想法是，我们必须从给定变量开始分支。 这就是常见的规则。 但这并不是一份详尽的清单。 还有更多 分数值接近 0.5 的变量就是一个规则。 在我们的问题中，变量对对象函数的影响最大。 在实际应用中，这可能会引起一些争议。 例如 我们正在最大化 x 1 加 10 x 2。 在这种情况下，影响最大的是什么？ 变量。 X 二是我们最大化的原因，其系数远远大于第一个决策变量的系数。 同样，没有证据表明，如果选择这些规则，方法一定会收敛得更快或更好。 但是，还是有不同的规则。 我们有一个默认值。 在大多数商用求解器中，比如简单方程，我们都有标准的默认配置。 而标准配置实际上是不同规则的混合。 好吧。 并没有一直使用的单一配方。 在大多数商业服务器中，我们都会在找到 LP 之前先进行预处理。 放松是一个预处理阶段。 我们并不是在讨论预处理，但预处理是非常常见的，预处理是为了尽量去除冗余。 这是第一件事，所以你现在不需要担心，就像在当今世界一样。 好吧，也许我们现实世界中的问题有一些冗余约束。 比如说，我们不需要担心这个问题，因为在预处理的时候，如果你有一个数字，如果这确实是一个不能忽略的信息，你会处理好的。 好的。 去掉多余的约束条件，开始使用更好的表述。 有了这个更好的表述，我们就可以开始使用分支边界了。 有了这个更好的公式，我们就可以开始相应的 LP 松弛，然后重新开始计算。 这就是为什么我不知道你是否还记得解决生产计划问题的那个问题。 你是说生产计划问题吗？ 我们解决的是整数公式、LP 松弛，然后我们使用 a。 还记得吗？ 是的，是的，我们两周前做过。 对于那些不记得的人来说，无论如何都要记得，我们曾经分析过这些解决方案的边界、差距和缺口，它们就像是可怕的一批，非常糟糕。 这说明了一些问题，因此，如果能研究出更好的线性松弛模型公式，可能会更好。 因为如果我们从一个非常差的线性松弛模型开始，当我说非常差的线性松弛模型时，我指的是一个线性松弛模型的最优值，从理论上讲，这个最优值与原始最优解相差甚远。 好吧，非常远，差距非常大。 但是，你看，我们开始根据 LP 松弛所提供的信息对变量进行分支。 如果这些信息都是垃圾，你能指望这条信息会花很长时间或更长的时间来找到一个好的解吗？ 我甚至没有提到最优解，因为对于大规模问题，我们并不一定需要找到最优解。 我们需要找到一个质量还可以、质量不错的解决方案。 一般来说，我不知道，也没有规则，但你说的最优解的 5%以内，是指下限和上限最终相差 5%。 如果这是一个大规模问题，那就是疯狂的问题，而不是你的情况。 所以我想要最优解。 我想要 0.00 的差距。 确保你能找到这些最优解。 因为你们都要解决一个 IP 问题，至少要解决一个 MIP 或 IP 问题。 因此，正确设置相对最优差距非常重要。 否则，你可能会觉得自己实际上是在分析最优解。 其实不然。 你分析的是一个可行的边界，但不是最佳边界。 好吧。 所以这一点非常重要。 所以，无论如何，我们有了这个标准，来探索例如未经证实的节点。 比如这个节点就是未经证实的节点。 假设在某一次迭代中，我有几个节点可以探索，为什么我说的是一个更大的问题呢？ 所以，主要的想法就是，在这里，更多的节点，等等等等。 在某些时候，我们需要确定如何继续利用这些节点。 你有不同的方法，因为最终你必须在第一个创建的节点或我不知道的、不那么遗憾的、不那么创建的节点或第一个节点之间做出选择。 所以你必须在最近的节点和最后的节点之间做出选择。 好的。 我们这里有不同的规则。 我们称之为最常见的规则。 第一种搜索策略。 基本上就是选择一个活跃节点。 活跃节点就是可以被探索的节点。 好的，它是最近的两个后代中的两个。 这种策略通常用于尽快找到原始边界或可行解。 因此，如果遇到可行性问题，要找到可行的解决方案是非常困难的，因为你可能会遇到两个问题。 一是很难找到问题的可行解。 第二，找到一个可行的解决方案非常容易，但要证明这个解决方案是最优的却非常困难。 这是一个不同的问题。 好吧，显然最终两者都有。 不管怎样，如果你的问题是要尽快找到一个原始可行解，以便更快地更新边界。 那么你应该使用深度优先搜索策略。 但我们还有其他策略。 例如，最佳节点优先策略包括选择活动节点，即以最佳对偶边界进一步探索的节点。 这种策略通常用于减少枚举树中需要评估的节点数量。 因此，假设我们知道节点数量过多。 因此，你希望减少需要探索的节点数量。 如果是这种情况，我们的想法就是选择下一个待探索节点的策略，即最佳策略节点。 而当他们说最佳二元约束时，正是你所想的，这就是你的情况。 你能找到对偶吗？ 基于对偶变量并利用这些信息的解决方案。 因此，对于所有节点，你最终可以选择哪些节点是哪些节点，而不是哪些节点需要探索。 我们所拥有的 正如我之前所说，我们所拥有的实际上是混合策略。 因此，如果你阅读过例如单纯形默认解算器、好解算器或所有常用于解决 LP 和 MIP 的解算器，我们所拥有的是解算器在开始解决给定问题时识别最佳策略的东西。 因此，这就解决了求解器的问题。 开始解决一个给定的问题。 要找到初始解非常困难。 所以，让我们先找到它，好了，我找到解了。 现在，我们切换到最佳节点优先策略。 因此，在算法过程中，并不存在唯一的关键策略。 在这些商业和求解问题中，它一直在变化。 所以有一种方法，我们不逐一挖掘，但有一种方法，比如，改变参数。 例如，你有一个给定的问题，出于某种原因，你知道在算法处理过程中应该首先使用这个问题。 因此，你可以在开始通过牙龈解决问题时加入这些信息。 例如，你可以提供以下信息。 请使用第一搜索策略。 因此，基本上你可以建立一个文本文件，其中包含所有分支边界信息的配置，以及你想在分支边界中改变的所有内容，改变默认策略。 该方法从你的最佳配置开始。 举例来说，这个配置比较复杂，但也可以这样做。 就是这样。 耶 有问题吗？ 很多。 那么，你已经得到了图形解法 再多说点关于单纯形的 再多说点关于分支边界的 好吧 你们还在吗？ 什么图形解法 这是非常基本的。 你基本上可以读到我们讨论过的所有内容。 简约法要高级一些，因为它需要一些我没讨论过的概念。 例如，基本解法。 我们已经讨论过了，但用的是非常快速的方法。 而分支边界的描述是非常基本的，最终的术语会有些不同，因为不同的教科书最终会使用不同的术语，例如： 终止节点或修剪节点 archibong 节点。 但原理是一样的。 好的。 我的建议是 我的建议是 尽可能多的在这个包容性的材料。 在最后的任务方面，显然我不是要求你们提供比我在这些幻灯片中提供的更多的内容。 好吗？ 所以，你们有附带的材料，但这并不意味着这就是给你们的。 如果你们想了解更多，可以学习。 这不是给我的。 这是给你们的。 好吗？ 如果你不想再学了，那也没关系。 但至少这些幻灯片的内容应该是没有的。 好的，应该学会 好的 这些，数量接近第五点。 怎么排序？ 就是这样 接近第五点的数字 这是要用的。 这很直观，因为这是离整数最远的数值，什么是离整数最远的数值？ 五点 九点接近于1. 1 接近于0. 2 接近于0. 3 接近于0. 4. 我也不知道零点一和如此这般的中间值是多少。 非常普遍的规则。 在我们的研究中，这个规则就是无论什么都会好。 所以，基本上这就是自动地在一般情况下，你说像我们认为的分支边界。 说我们在这里的问题中使用了这个特殊的东西。 我试图联系起来，我们是如何在大脑中首先使用这一点的，以及。 如何使用。 所以，基本上我们必须进行分析。 例如，假设你正在执行这个算法。 比如说 这就像一个四舍五入算法。 所以你要分析，你有一个给定的值，你必须只分析给定解的小数部分。 你要分析它是否更接近 5，比如，你是否从它开始分支，或者，你是否有若干小数部分，你是否有一个小数值列表。 我不知道这是不是你的问题。 你可以选择小数部分接近 0.5 的变量。 但是，内部信息是自动生成的。 我的意思是，它可以在算法的任何步骤中出现。 好吧，它甚至可以像读数或勒克斯法那样，对解法的分数部分施加进一步的约束，诸如此类。 没问题了 好的 太神奇了 好的，太棒了 耶，耶！ Yay, yay！ 非常感谢 Thank you very. 非常感谢 玩得开心 你有整个星期。 这个 我们能谈谈我们的谈话吗？ 我们能谈谈我们的话题吗？ 好啊 很快，因为我累了。 你说吧 你的小组在哪里？ 你的小组呢？ 好吧 我能用 我可以使用。 可以 Yes. 也可以 这就像一个婚礼 我们有很多的后卫， 所以我们尝试重新开始。 你有什么。 婚礼 好吧 婚礼 所以我们尽量把不同的人分成不同的组 这样他们就能坐在同一张桌子上了 这就像一张桌子的最大率，像一个人。 然后我们试着感谢你。 他们