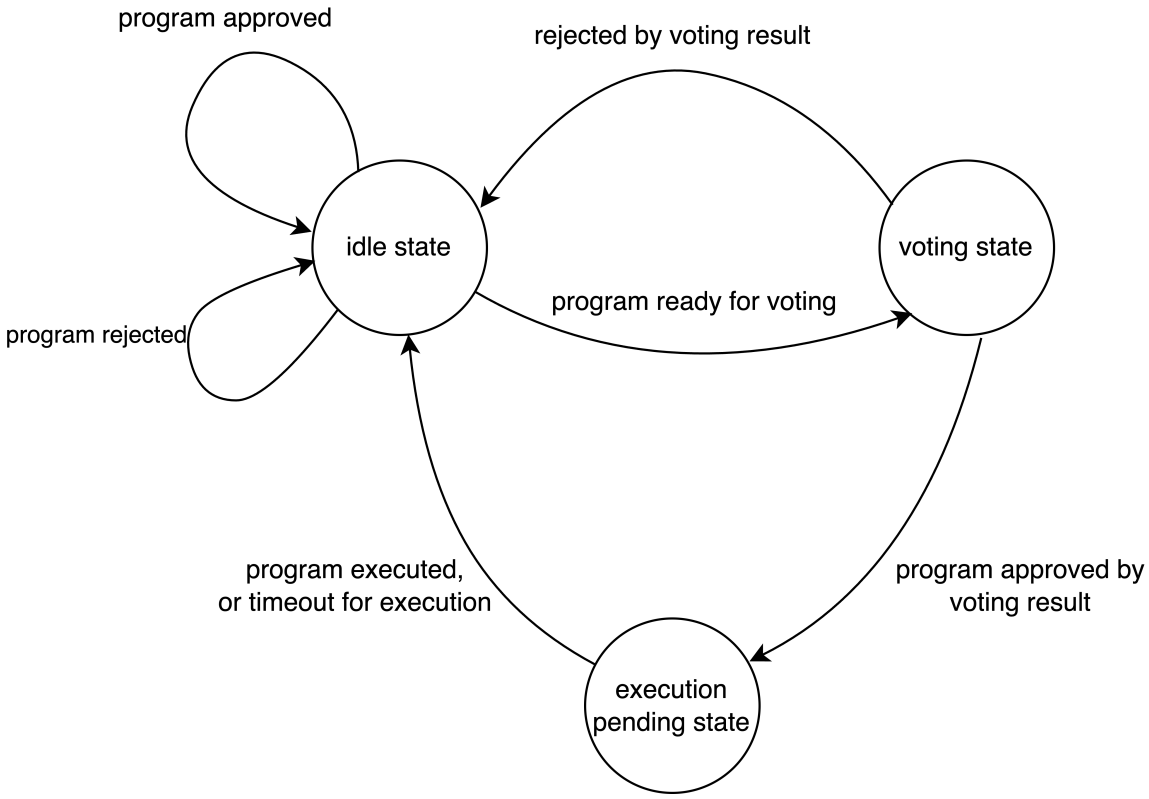

图 1: DARC 的有限状态机

1 投票

1.1 投票过程和状态

如图 ?? 所示，您可以将每个 DARC 协议视为一个有限状态机（FSM），它具有三种状态：

1. 空闲状态：在此状态下，DARC 协议可以接受任何程序。当程序被接受并执行后，DARC 协议转回到空闲状态。同样，如果程序被拒绝，DARC 协议返回到空闲状态。当程序包含需要投票的操作时，DARC 协议转换到投票状态。

2. 投票状态：在 DARC 协议的投票状态期间，所有用户被允许且只被允许执行包含单一投票操作的程序。如果程序包含多个操作，程序将被拒绝。同样，如果程序包含的单一操作不是投票，程序也将被拒绝。如果当前投票被批准，系统将自动转换到等待执行状态。此外，投票状态有一个最小时间限制。如果在此时间限制内，每个投票项的支持票数未总体超过定义的阈值，当前的投票轮被视为不成功，系统将自动返回到空闲状态。只有当所有投票项在指定时间限制内获得批准时，程序才会被 DARC 协议批准。

3. 等待执行状态：当一个程序被 DARC 协议投票并批准后，它可以在等待执行状态中被执行。等待执行状态也有一个最小时间限制。如果程序在此限制内被执行，它将被成功执行，系统将自动恢复到空闲状态。如果执行没有在此限制内发生，程序将失败，DARC 协议将丢弃程序，自动返回到空闲状态。

1.2 投票规则

每个投票规则由以下项目组成：

- 投票代币类别列表：允许为此投票项投票的代币类别索引列表。它包含至少一个有效的代币类别索引号。列在此数组中的所有代币都被视为有效投票代币。持有任何包含在此数组中的代币的所有代币持有者都被允许为此项投票。
- 批准阈值百分比：以百分比表示的批准阈值。

- 布尔标志 bIsAbsoluteMajority：一个布尔标志，指示投票模式是绝对多数还是相对多数。
- 投票持续时间（秒）：此投票过程的最大持续时间，以秒为单位。如果投票过程超过此参数的时间，投票过程将被终止，投票结果将被设置为失败。
- 执行等待持续时间（秒）：在投票过程结束后，操作者执行批准程序的最大持续时间，以秒为单位。如果程序被批准，操作者必须在定义的持续时间结束前执行此程序；否则，程序将被中止，DARC 将重置为空闲状态。
- 布尔标志 isEnabled：一个必须设置为 True 的布尔标志，如果投票规则被成功初始化并启用。
- 注释：存储在投票规则中的字符串，包含此投票规则的额外信息、评论或外部 URL。

```
struct VotingRule {

  /**
   * 投票代币类别索引列表
   */
  uint256[] votingTokenClassList;

  /**
   * 投票政策的批准阈值百分比
   */
  uint256 approvalThresholdPercentage;

  /**
   * 投票政策的投票持续时间（秒）
   */
  uint256 votingDurationInSeconds;

  /**
   * 投票政策的执行等待持续时间（秒）
   */
  uint256 executionPendingDurationInSeconds;

  /**
   * 投票政策是否启用
   */
  bool isEnabled;

  /**
   * 投票政策的注释
   */
  string note;

  /**
   * 投票政策是绝对多数还是相对多数。
   */
  bool bIsAbsoluteMajority;
}
```

1.3 投票类型

通常，在 DARC 协议的投票系统中，有两种类型的投票方法：

- 绝对多数：在绝对多数投票中，批准需要超过一个固定阈值，以总投票权重的百分比表示。如果批准票的组合权重超过预定阈值，投票被认为是成功的。例如，总权重为 1000，阈值设置为 70%，如果批准票的总权重超过 700，投票被认为是成功的。

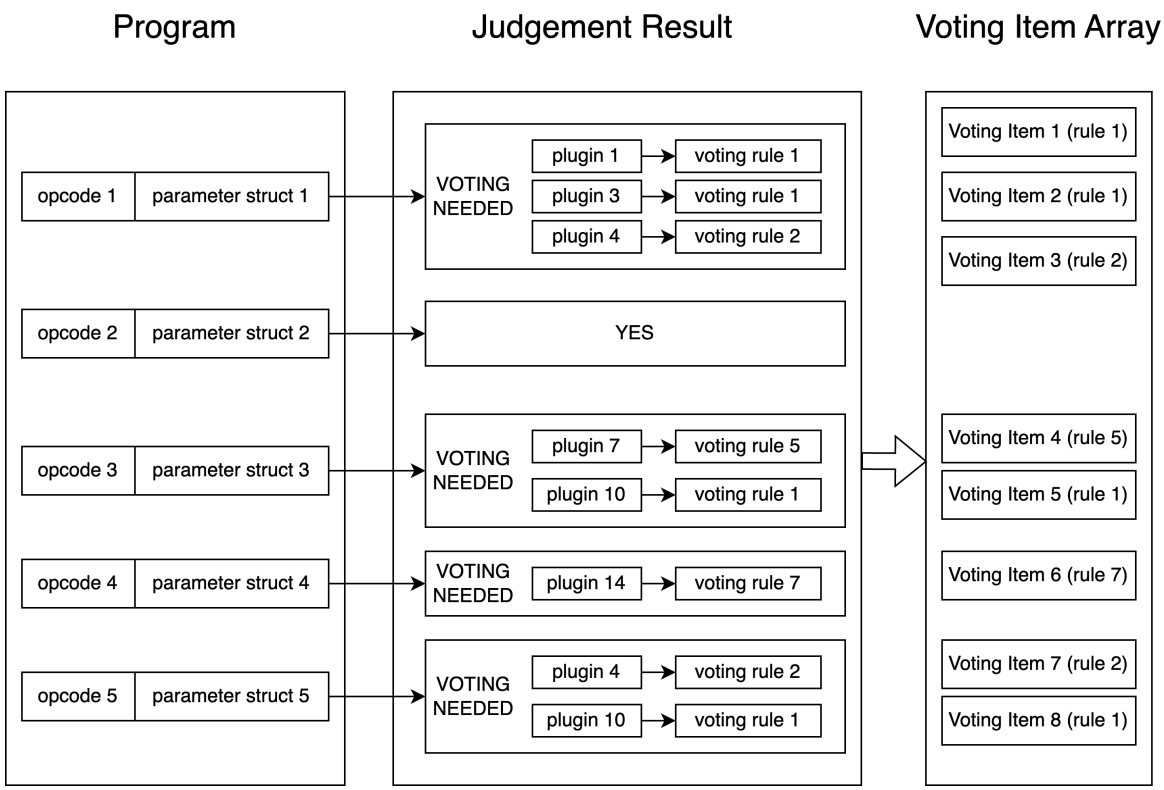

图 2: 投票规则和项

- 相对多数：在相对多数投票中，批准是相对于总投票权重的百分比。与绝对多数不同，总投票权重是实际投票权重的百分比，而不是预定义的绝对值。例如，如果总权重为 1000，但只有 300 权重在投票中被投出，相对多数要求批准的投票权重超过投出票的 70%（$0.7 * 300 = 210$），投票才被认为是成功的。

我们提出“绝对多数”和“相对多数”投票选项，是因为这两种模式在平衡投票结果时提供了不同的侧重点。

关于绝对多数，其优势在于批准的投票权重只需要占总可能投票的一部分，提供更多灵活性。它适用于重大决策的情况，其中一个显著多数足以确定结果，以促进更广泛的共识。其应用场景适合于重要决策，其中至关重要的是获得大多数人的支持。使用绝对多数作为标准有助于促进更广泛的共识。

相比之下，相对多数的优势在于批准的投票权重只需要构成当前投票中的最大份额，提供了更高的便捷性和灵活性。这种方法适用于相对多数足以确定结果的情况。它的适用性扩展到需要更灵活和快速决策过程的情况。相对多数有助于在不需要固定百分比的总可能投票权重的情况下更迅速地达成共识。

通过提供绝对和相对多数选项，我们增强了投票规则以满足不同协议和组织的多样化治理需求的适应性。选择适当的投票模式有助于更好地满足特定决策场景的共识和效率需求。

1.4 投票机制

对于每个程序，在收到来自操作前插件的批准并在沙箱内完全执行后，由所有操作后插件引导的判断系统，评估每个操作。如果对特定操作的所有插件的最高级别判断标记为“VOTING_NEEDED”，则收集与该插件相关的所有投票规则。一旦判断系统汇编了程序中每个操作对应的所有投票规则，它会生成一个投票项数组，为其各自的条目分配每条规则。

对于每个操作，可能同时触发一个或多个不同的插件，所有这些插件的返回类型为“VOTING_NEEDED”。在这种情况下，将按顺序收集指向该操作的每个插件的投票规则。另一方面，对于每个程序，不同的操作可能会被同一个插件依次触发。在投票项数组中，每个操作的对应投票规则将生成一个新的投票项。

图 **??** 是一个示例，说明 DARC 协议是如何基于判断结果初始化投票项数组的。

一旦投票项数组被初始化，作为有限状态机，DARC 协议转换到投票状态。假设在投票项数组中有 N 个投票项，所有操作者都将被允许投票，提交一个长度为 N 的布尔数组。每个布尔值代表操作者对相应投票项的支持或反对。对于每个操作者，在每个投票状态中，他们可以且必须只投票一次，确保投票操作中的参数是一个长度为 N 的布尔数组。如果操作者尝试执行超过一次的投票操作或提供一个长度不为 N 的布尔数组，投票程序将被自动拒绝。

在 DARC 协议中，每个投票项都有两个计数器：一个用于“YES”票，另一个用于“NO”票。每当一个操作者执行有效的投票操作时，每个投票项都会计算该操作者在相应投票规则中的总投票权重。在给定的投票项中，每个投票规则定义了一组允许的代币级别 $C = [c_1, c_2, ..., c_n]$，其中 n 是代币级别的数量。每个级别 c_i 有一个相应的投票权重，表示为 w_i。假设在投票过程中，一个操作者选择对这个投票项进行投票，操作者的投票权重 W 通过以下公式计算：

$$W = \sum_{i=1}^{n} w_i v_i$$

在此公式中，W 代表操作者对此投票项的总投票权重，求和遍历所有允许投票的代币级别 c_i。对于每个级别 i，v_i 是此操作者在该级别拥有的代币数量。

在每个操作者提交有效的投票操作后，每个投票项将把相应的投票权重增加到 YES 或 NO 计数器中。如果操作者在可投票代币集合中持有零代币，对于特定的投票项，既不会影响 YES 计数器也不会影响 NO 计数器。在投票过程完成后，DARC 协议将根据以下标准确定投票是否结束：

1. 如果所有投票项中至少有一个在绝对多数模式下运行的投票项收到足够数量的 NO 票，整个投票过程将提前终止，导致投票失败。随后，DARC 协议转换到空闲状态。例如，如果一个投票项的批准阈值为 66%，以绝对多数模式运行，并且在截止日期前收集到超过 34% 的 NO 票，整个投票过程将立即结束，导致投票失败并转换到空闲状态。

2. 当所有投票项仅在绝对多数模式下运行，并且每个投票项都获得了超过批准阈值的 YES 票时，整个投票过程将提前结束，表明投票成功。随后，DARC 协议转换到等待执行状态。

3. 当投票状态不符合标准 1 和 2 时，DARC 协议将等待截止日期到期进行评估。如果在截止日期后，每个在绝对多数模式下的投票项都收到了等于或超过总投票权重乘以阈值的 YES 票，并且对于每个在相对多数模式下的投票项，该项获得了等于或超过提交的投票权重乘以阈值的 YES 票，投票被批准，DARC 协议转换到等待执行状态。如果任何投票项未能满足此标准，投票被认为是不成功的，DARC 协议恢复到空闲状态。

在进入等待执行状态后，当前投票状态中存储的程序已被批准且准备执行。程序必须在新的等待执行截止日期前完成。成功执行后，DARC 协议转换到空闲状态。然而，如果在执行过程中发生错误或异常，或者如果在截止日期前没有人执行程序，DARC 协议也会恢复到空闲状态，并且程序将无法继续执行。