Описание Менеджера эвристик

Пусть дан большой аналитический запрос с внутренними соединениями. Для его планирования будем выполнять следующие шаги:

1. Представим запрос в виде набора связных компонент, любые две таблицы из разных компонент не имеют условия соединения между собой.

2. На основе количества связных подграфов введём функцию бюджета в компоненте. Если его достаточно, то можно позволить более долгое время оптимизации и более оптимальные эвристики/алгоритмы.

3. Для каждой связной компоненты будем выполнять итеративную декомпозицию на топологии и их планирование в цикле, пока не получим план для компоненты. Если что-то получилось выделить, возвращаемся в начало цикла, если нет, то переходим на планирование выделенных топологий и снова в начало цикла:

   (a) Выделим из компоненты **плотные графы** (с 4+ вершинами) с плотностью от **density_border**

   (b) Выделим циклы.

   (c) Если после выделения плотных графов и циклов, компонента остаётся связной, то выделим дерево, иначе выделяем звёзды и цепи.

   (d) Выделим звёзды с лучами длины до **ray_length**.

   (e) Выделим цепи.

   Звездой это группа вершин: центральная и связанные с ней цепи, каждые две цепи не связаны с друг другом в оставшейся топологии. Центральная вершина имеет количество строк в 10+ раз больше чем у 2+ соседей или соседей от 3+.
   Все выделенные топологии отсортируем по количеству связных подграфов в топологии.
   Распределим бюджет компоненты по топологиям в соответствии со сложностями. Каждая топология после планирования становится вершиной, получаем новый граф.
   Пусть **border** - это количество свзяных подграфов, больше которого топология считается сложной, меньше лёгкой.
   Планирование:

   (a) Если бюджета для текущей топологии не осталось, то планируем эвристически, иначе смотрим на сложность. Для лёгкой топологий используем встроенный ДП, для тяжёлой эвристику. Эвристики:

      i. Цепь: алгоритм GOO, и пока остаётся бюджет будем оптимизировать деревья размера до **k**.

      ii. Цикл: найдем соединение с наибольшей кардинальностью, и разобьём цикл по этом соединению - уберём одну таблицу в этом соединении. Спланируем цепь, и присоединим удалённую таблицу.

      iii. Звезда: сначала спланируем лучи как цепи, затем будем последовательно присоединять к центру использую GOO (по кардинальности) с центральной фиксированной вершиной.

      iv. Плотный граф: GOO с оптимизациями деревьев размера до **k**.

      v. Дерево: если достаточно бюджета, то используем IKKBZ/linDP, иначе GOO с оптимизациями деревьев размера до **k**.

4. Объединим планы компоненты запроса алгоритмом GOO(по стоимости) с помощью декартового произведения.