Алгоритм решения

ICCAD contest 2015

Шаг 1. Набор статистики (поиск соответствий)

- Выбираем некоторое подмножество $V^* \subseteq V_{\Sigma'} \times V_{\Sigma''}$ всех пар вершин в заданных схемах (есть несколько стратегий).
- Для каждой пары $(u, v) \in V^*$ ищем существующие соответствия.
  - Поиск осуществляем
    - до первого соответствия
    - или пока не набрали k-соответствий
    - или исходя из некоторого бюджета времени.
  - Если соответствия нет или кончилось время переходим к следующей паре.
  - Порядок просмотра пар зависит от стратегии выбора V^* (см. далее)

Шаг 1а. Выбор подмножества пар вершин

- Если размер схем позволяет, то выбираем все пары.
- Иначе выбираем пары исходя из понятия окрестности вершин (ввести понятие окрестности можно по разному – процесс творческий). Предлагаю следующий рабочий вариант (можно критиковать):
  - Пусть изначально заданы опорные пары: пары вершин согласованных входов и выходов
  - Будем считать, что пара (u,v) находится на расстоянии не более k от пары (m,n), если между u и m существует путь длины не более k в схеме Σ' и соответственно путь длины не более k между v и n в схеме Σ''.
  - Тогда выбираем некоторую опорную пару и просматриваем все пары на расстоянии не более k в ее окрестности. Все найденные пары помечаем как опорные. После просмотра помечаем пару как просмотренную, переходим к следующей паре.
  - В конце все просмотренные опорные пары выбираем в качестве искомого множества пар.
- Параметр k регулирует окрестность просмотра. Обычный поиск в ширину или глубину должен сработать.
- Предложенный способ выбора множества пар не накладывает никаких ограничений на порядок их просмотра при поиске соответствий
- Если есть другие идеи как выбрать хорошее множество пар – смело реализуйте.

Шаг 1б. Поиск соответствий

- Берем произвольную пару вершин из полученного на предыдущем этапе множества пар вершин
- Для каждой вершины строим все разрезы ширины k (алгоритм можно найти здесь http://cadlab.cs.ucla.edu/~cong/papers/fpga99_cut.pdf).
- Просматриваем разрезы в порядке возрастания ширины. Берем разрез заданной ширины для первой и второй вершины. Если такой пары разрезов нет, то увеличиваем ширину на 1 и переходим к следующей паре.
- Для выбранной пары проверяем эквивалентность порожденных ими конусов (алгоритм проверки зависит от ситуации: использовать abc, Boolean matching, симуляцию подсхем на наборах и т.д.). Так как вариантов много, эту часть надо спроектировать так, чтобы можно было потом подключить другие алгоритмы. Так как времени мало, то нужно сейчас использовать самый простой рабочий вариант.
- Найденные соответствия сохраняем.
- Переходим к следующей паре.
- Если для заданной пары не удалось найти соответствия, то формально помечаем пару как эквивалентную (даже если

Шаг 2. Поиск решения по набранной статистике – поиск покрытия

- Решаем задачу при помощи динамического программирования. Для начала рассмотрим случай, когда мы хотим получить разбиение только на эквивалентные пары подсхем.
- Так как я не прорабатывал детали и алгоритм является адаптацией классических алгоритмов mapping-а, то тут могут быть ограничения применимости, связанные со структурой схемы и тем фактом, что по сути мы строим покрытие.
- Находим топологический порядок вершин в первой и второй схеме.
- На основе этих порядков упорядочиваем вершины выбранного множества вершин, для которого мы нашли соответствия. Есть вопрос, можно ли всегда построить такой порядок (я над этим моментом еще не думал). Возможно, что можно построить только частичный порядок, а несравнимые элементы придется просматривать в произвольном порядке с возможной потерей оптимальности и каких-то промежуточных решений (или вообще решения целиком).
- Пусть нам удалось получить топологический порядок на множестве выбранных пар вершин (или какой-то частичный топологический порядок). Пусть для первых N вершин мы нашли решение задачи. Будем считать, что каждой паре вершин приписана стоимость частичного решения: вектор размеров подсхем.

Шаг 2. Поиск решения по набранной статистике – поиск покрытия

- Рассмотрим N+1 пару вершин (для частичного порядка, скорей всего, придется просматривать целое множество вершин).
- Для этой вершин просматриваем все найденные соответствия в порядке возрастания размеров конусов.
  - Если в множестве пар соответствующих входов рассматриваемого соответствия есть неэквивалентная пара (для нее не найдено соответствие, она вообще не просматривалась), то переходим к следующему соответствию.
  - Иначе по всем парам входов рассчитываем суммарную стоимость частичного решения (стоимость решений во входах + стоимость рассматриваемого соответствия). В общем случае, возможны конфликты решений во входах (есть две стратегии – наложить ограничения(на структуру анализируемых схем и соответствия, которые мы ищем), чтобы конфликтов вообще не было, либо уметь проверять на конфликты и отбрасывать соответствие, если есть конфликты).
  - Если нашли решение лучше, то сохраняем стоимость найденного частичного решения (и само решение , то есть фиксируем соответствие для просматриваемой пары), переходим к следующему соответствию.
  - Когда для заданной пары вершин все найденные соответствия пройдены, то мы находим наилучшее частичное решение среди найденных соответствий.
  - Переходим к следующей паре в соответствии с найденным порядком.

Шаг 2. Поиск решения по набранной статистике – поиск покрытия

- Предложенный набросок алгоритма можно расширить, чтобы учитывать возможную неэквивалентность подсхем
- На начальном этапе можно отказаться от оптимизации, просто пытаться таким образом строить хоть какое-то разбиение
- Так как это еще не до конца проработанная идея, то в предложенном варианте может быть много косяков. Буду рад услышать критику, замечания и какие-то Ваши идеи : как решать задачу.
- Вариант с рисунками будет позже.