|  |
| --- |
| Попробовать сделать расчет производных "отложенным"  это позволит отказаться от учета переменной производной  в матрице и сделать значение производной не переменной,  а просто значением - константой на итерации.  Такой подход можно попробовать и для регуляторов, и для  частоты в узле.    На самом деле эта производная уже есть в нордсике в виде произведения на текущий h.  Описание: "On Angle References in Long-Term Time-Domain Simulations"  Davide Fabozzi, Thierry Van Cutsem  IEEE TRANSACTIONS ON POWER SYSTEMS, VOL. 26, NO. 1, FEBRUARY 2011 |
| В BuildEquations/BuildRightHand для узлов сделать  объединение по циклам по ветвям и по генераторам,  чтобы не мотать два цикла при построении матрицы |
| В пользовательскую DLL ввести номер версии,  И если он меньше, чем текущая версия движка,  Требовать наличия исходника и компилировать заново. Необходимо для контроля изменений движка (количество уравнений блоков, например)    Внутрь ДЛЛ можно даже поместить исходный код или графическое описание модели, чтобы не требовать наличия файла |
| В компиляторе для переменных, которые не являются переменными состояния (значение+индекс), но являются переменными (значение, которое можно читать и записывать), приходится делать проверку индекса на значение DFW2\_NONSTATE\_VARIABLE\_INDEX. Подумать на тему как дать компилятору информацию о типе переменной заранее, чтобы избавиться от проверок. |
| Проверить ProcessDiscontinuity в генераторах. Для мустанговской модели уравнения расчета отличались от уравнений модели, что приводило к неправильной работе ограничений, в частности от Eq. |
| Подумать над тем, как должен выглядеть Lag с переменными ограничениями. Похоже критерий для снятия с ограничения по знаку производной подходит не очень. Пример - скачки на ограничениях по току и напряжению в АРВ Мустанг |
| В Init или ProcessDiscontinuity можно ввести список девайсов, запрашивающих DiscontinuityRequest. При зависании можно будет определить какие девайсы входят в алгебраический цикл. |
| При ограничении шага после отказа корректора можно анализировать наличие предстоящих событий и характеристики сходимости усредненные для пройденного времени. Если событий нет и сходилось хорошо - можно увеличивать шаг чаще |
| В KLU complex судя по всему оптимизатор изменяет порядок вычислений, поэтому debug и release дают разные результаты. Сделать их идентичными позволяет включение оптимизации в debug  При интегрировании процессов с большими изменениями естественно падает Ньютон. Улучшает ситуацию выбор шага. При этом шаг Ньютона лучше выбирать с минимальным ограничением пропорциональным шагу интегрирования, а не 0.1, как выбрано сейчас. Замечено, что Ньютон сходится до 1E-10, но ошибка относительно предиктора не дает закончить шаг. Возможно на шагах близких к минимальному надо принимать решение Ньютона и формировать Нордиск по нему, игнорируя тем самым предиктор (как бы ошибка). И наконец проверить теорию, что завалы Ньютона могут быть обусловлены периодикой углов: если приращение угла > - то оно может быть отрицательное (?) |
| Для полярных координат попробовать сделать проверку корректора в нормализованных углах - иначе при больших отклонениях от начальных условий значительное абсолютное значение может искажать проверку по Rtol. Для этого нужно в abs(V)\*Rtol+Atol подставить wrapped угол. Знак неважен, естественно. |
| InitExternalVariables вроде бы не нужны нигде, кроме Init и InitNordsiek. Проверить !  Сделать проверку/простановку DS\_ABSENT для всех девайсов. Возможно единую для всех CDevice |
| При инициализации реле с выдержками времени регистрируют себя на событие даже тогда, когда приводятся в нормальное положение (например, реле выхода из форсировки). Надо продумать, как избавиться от лишних событий. В результате дум решено инициализировать реле значением по текущему значению и уставке. При инициализации не вводить выдержки времени, предполагая, что входной параметр и уставка бесконечно долго были в заданном начальными условиями отношении. |
| При компиляции модели нужен парсер, который выделит из выражений для расчета констант дерево постоянных параметров, получаемых из таблиц. |
| Адресация параметров - тройкой. Продумать умолчания, при которых параметр ищется в связанных устройствах ниже или выше, если не указаны таблица и идентификатор.  Продумать как адресовать ветви. |
| В компиляторе нужна инициализация #base. Для автоматики это лучше всего сделать присваиванием от внешней переменной в Init. В ProcessDiscontinuity #base нужно игнорировать - их значения не меняются процессе расчета. |
| При сжатии блока использовать RLE всегда. Тип блока "RLE" убрать. Попробовать реализации 7bit RLE с повторами/пропусками (compression.ru) или 3-байтный (fileformat) |
| В производной нужно выяснить как выполнять реинициализацию при скачке на входе. Вариантов 2 : подгонять запаздывание или рассчитывать выход при фиксированном запаздывании. В последнем случае на выходе происходит скачок K/T\*dX в виде одиночного пика. |
| Для подавления рингинга Адамс-2 можно в случае нескольких удачный шагов (10) для которых невозможно увеличить шаг попробовать сделать 1 шаг BDF по дифурам, после чего снова перейти на Адамс (см GoodStep() и UpdateNordsieck()) (статья  A NEW STABILIZATION OF ADAPTIVE STEP TRAPEZOID RULE BASED ON FINITE DIFFERENCE INTERRUPTS). А еще можно попробовать пересчитывать производную Адамса по выражению BDF (1.7) из статьи. Для этого есть Tminus2Value |
| При поиске зеро-кроссинга на втором порядке похоже нужно в любом случае решать квадратное уравнение, чтобы не пропустить двойное изменение. Сейчас сделан поиск времени зерокроссинга только в случае, если изменился знак функции зерокроссинга. Для первого порядка это работает, но для второго порядка есть вероятность пропуска временного зерокроссинга. Можно попробовать вне зависимости от изменения знака искать корни функции зерокроссинга. |
| Для решения задачи при наличии задержек есть мысль сохранять полный нордсик на глубину задержки. Нордсик позволит точно получить значение в |
| В контроле ZC сделать одновременный контроль знака и наличия корней. Иначе либо будем зависать, либо пропускать. |
| Написать комментарии про Skip постоянных элементов матрицы |
| В ZC можно сделать опцию поиска пересечения только по изменению знака или с предварительным поиском корня. Без поиска корня для всех пересечений должно работать быстрее, но с вероятностью пропуска. Если ZC будет более или менее централизован опцию сделать легко. Можно использовать теорему Штурма для определения количества корней на интервале шага. |
| Профайлер показывает что самая долгая функция из локальных – atan2. Используется она в расчете угла в узле, после чего угол еще дифференцируется для получения скольжения. Можно поискать более быструю апрроксимацию atan2, но из того, что уже нашлось погрешности около 0.5 градусов, что для Ньютона неприемлемо. Другой вариант состоит в отказе от расчета угла и скольжения в шаге интегрирования при условии, что в узле нет ничего, что использует угол или скольжение. Эксперимент со схемой СЗ показывает, что отказ дает время расчета 4.8с. Если считать для всех узлов – 5.8с. Определить нужно или не нужно в узле считать угол и скольжение можно с помощью перекрытой GetExternalVariable, в которой поставить флаг или вообще подменить указатель на реализацию узла. Проблем три: 1. Суперузлы (флаг должен быть в суперзузле), 2. Возможность массового обращения к углам/скольжениям от автоматики в большинстве узлов, 3. Результаты расчета (их можно досчитать после расчета для каждого нужного канала). |