Применимость методов машинного обучения для тестирования моделей микропроцессора

Н. А. Гревцев[1], А. Д. Манеркин[2], П. А. Чибисов[3]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, ngrevcev@cs.niisi.ras.ru;

[2]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, manerkin@cs.niisi.ras.ru;

[3]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, chibisov@cs.niisi.ras.ru

Аннотация. В статье приведен обзор использования методов машинного обучения для различных направлений функциональной верификации. Рассматривается использование машинного обучения в «pre-silicon» верификации, а именно в имитационном тестировании и верификации при помощи UVM. Приводится обзор в области «post-silicon» верификации. Делается вывод об основных областях применения машинного обучения, а также о возможных будущих направлениях исследований.

Ключевые слова: верификация микропроцессоров, машинное обучение, Deep Learning, UVM

1. Введение

В связи с возрастающей сложностью современных микропроцессоров всё больше времени уделяется их верификации. На нее может приходиться до 70% времени создания процессора. В результате верификация становится крайне ресурсозатратным этапом проектирования. Более того, усложнение микроархитектуры современных цифровых устройств приводит к тому, что анализ результатов верификации вручную становится все менее эффективным, а ручное написание и запуск случайных тестов часто является избыточным.

Разрабатываются все новые методы стандартизации и автоматизации тестового окружения. В 2011 году был выпущен единый верификационный стандарт проверки цифровых схем – UVM (Universal Verification Methodology), позволяющий за короткие промежутки времени настраивать стандартную тестовую среду, и впоследствии многократно ее использовать, к примеру, для различных версий проекта.

В последнее время широкое использование в области создания и тестирования программного обеспечения получили методы машинного обучения. Вслед за областью программного обеспечения проводятся все больше исследований на тему применимости методов машинного обучения в области верификации аппаратного обеспечения. Применение данного набора технологий является перспективным направлением для ускорения, повышения качества и эффективности проверки цифровых схем.

В главе 2.1 данной статьи разбирается применение методов машинного обучения для «pre-silicon» верификации микропроцессоров в целом, в главе 2.2 приведен краткий обзор использования машинного обучения в верификации при помощи UVM, в главе 3 рассмотрено их применение для «post-silicon» верификации. В заключении приведены выводы по результатам научных работ в данной области и применимости машинного обучения для задач верификации.

2. Использование машинного обучения в «pre-silicon» верификации

Стадия pre-silicon верификации, то есть до выпуска разрабатываемого микропроцессора, является основным этапом верификации, так как именно на этом этапе разработка проекта осуществляется параллельно с его верификацией, а стоимость исправления ошибок минимальна. Поэтому большая часть работ, посвященная применению методов машинного обучения в верификации, направлена на различные этапы данной стадии.

2.1. Simulation-based тестирование

Ввиду сложности современных микропроцессоров при их проектировании неизбежно возникают ошибки. Ошибки могут нанести серьезный ущерб и привести к высоким издержкам. При этом их исправление в уже изготовленных микросхемах, как правило, невозможно. Поэтому необходимо выявлять как можно больше ошибок перед изготовлением процессора, еще на этапе проектирования.

Для этого используют различные методы функциональной верификации моделей микропроцессоров, нацеленные на обнаружение ошибок, то есть отклонений поведения модели от требуемого. Наиболее широко используемый на практике метод верификации микропроцессоров – имитационное тестирование (simulation-based

verification). Он состоит в программной имитации работы микропроцессора, описываемого моделью, с помощью симулятора в рамках ряда специально разработанных тестовых сценариев, представляющих основные варианты использования функций этого микропроцессора, возможные при его эксплуатации. Другим методом является формальная верификация (*formal verification*). Этот способ заключается в формальном доказательстве соответствия модели микропроцессора заданному набору свойств при всевозможных сценариях взаимодействия с ним. Оба метода имеют как достоинства, так и недостатки, и нередко для верификации микропроцессора используют как тестирование, так и формальную верификацию [1].

Исходя из теоретических предпосылок, случайные воздействия способны реализовать все возможные комбинации при наличии достаточного времени, но на практике при верификации очень сложных устройств случайный подход имеет трудности в своевременной реализации всех комбинаций. В результате, часто требуется направлять тестовую среду с целью реализации конкретных состояний. Такой ограниченно-случайный подход является достаточно мощным средством, однако он опирается на обширный опыт управления тестовой средой для достаточной верификации проекта. С увеличением сложности устройств, управление средой становится все более трудоемким и затратным по времени. Это приводит к тому, что верификация, способная охватить все точки покрытия, становится главным фактором, влияющим на длительность создания устройства [2].

Машинное обучение может значительно улучшить средства достижения необходимого покрытия, предоставляя механизм, с помощью которого можно отслеживать уровень покрытия, управляемые параметры верификации, а также изучать и затем улучшать взаимодействие между ними.

Предложенный алгоритм заключаются в следующем:

1) Для проекта выполняется ряд симуляций, в которых параметры моделирования тестируемого устройства, теоретически способные повлиять на покрытие, изменяются случайным образом.

2) Для каждого отдельного запуска моделирования регистрируется следующий набор входных данных верификации:

- параметры верификации, управляющие входными параметрами;
- параметры тестовой среды;
- параметры конфигурации устройства;
- результаты покрытия, в которых перечислены все состояния функционального покрытия и указано, было ли конкретное состояние достигнуто или нет во время моделирования.

3) Результаты всех симуляций затем обрабатываются алгоритмом машинного обучения, который отслеживает результаты покрытия в зависимости от входных данных.

4) В ходе серии симуляций алгоритм машинного обучения узнает, какие комбинации подмножеств входных данных проверки необходимы для достижения каждого состояния функционального покрытия. Список состояний функционального покрытия, отслеживаемых алгоритмом ML (*Machine Learning*), может быть отфильтрован с целью исключить регулярно встречающиеся состояния функционального покрытия, чтобы сосредоточиться на чрезвычайно редких и труднодостижимых случаях.

5) После обучения алгоритм машинного обучения генерирует набор рекомендаций для входных данных, тем самым увеличивая вероятность попадания в непокрытые состояния, ускоряя процесс разработки проекта.

Одним из примеров реализации подхода является работа [3], которая делает акцент на создание эффективных наборов входных данных с целью увеличения покрытия. Предлагается основанный на ошибке реконструкции метод анализа особенностей наборов данных с помощью глубокого обучения и нахождения таких наборов данных, которые затрагивают труднодостижимые точки покрытия. Экспериментальные результаты показывают, что предлагаемый метод является эффективным в нахождении эффективных наборов данных для моделирования.

Важным направлением работ является сравнение различных методов машинного обучения [4]. Автор работы предлагает новую методологию создания модели с использованием SVM (машины опорных векторов), классификатора повышения градиента и нейронных сетей, нацеленную на замену генерации случайных тестов ускоренным сбором покрытия. При использовании данной методологии авторам удалось добиться достижения целевых показателей покрытия при помощи меньшего количества тестов. Также было обнаружено, что нейронные сети обеспечивают лучшие результаты по сравнению с классификатором повышения градиента и машиной опорных векторов.

Многие авторы фокусируются на использовании конкретных подходов для решения задачи генерации тестов и воздействий. Авторы работы [5], полагая генетические алгоритмы лучшим способом генерации последовательностей входных данных, способных достичь желаемого уровня охвата, разработали и протестировали с использованием трех различных проектов не-

сколько подходов, базирующихся на генетических алгоритмах. Во всех ситуациях процент наборов воздействий, сгенерированных с использованием высокопроизводительных генетических алгоритмов, был выше значений, полученных при использовании случайного моделирования. Вдобавок, в большинстве случаев подход на основе генетических алгоритмов достигал большего значения покрытия за тест в сравнении с результатами случайного моделирования. Эксперименты подтвердили, что в большинстве случаев генетические алгоритмы способны превзойти случайную генерацию воздействий, которая используется в классическом способе проведения верификации, учитывая заполнение уровня покрытия за один тест.

Существуют также исследования, развивающие подходы на основе обучения с подкреплением. Обучение с подкреплением является перспективным методом машинного обучения, способным самостоятельно адаптироваться, применяется для верификации широкого спектра устройств. Работа [6], основываясь на более ранних исследованиях в данной области, предлагает универсальный метод, который уменьшает усилия по созданию функциональных тестов, предлагая возможность их повторного использования для верификации различных RTL-проектов. Новая методика использует обучение с подкреплением при помощи модели «актер-критик» для создания тестовых наборов данных для различных RTL-проектов. Были проведены эксперименты, сравнивающие подход на основе модели “актер-критик” с подходом, когда тестовая система не ограничивает случайные воздействия.

Результаты экспериментов показали, что предложенный метод позволяет увеличить долю сработавших точек покрытия на величину от 11,4% до 50% в зависимости от тестируемого устройства. В случае, если оба подхода достигали 100% покрытия, подход на основе модели «актер-критик» позволял значительно уменьшить время достижения 100% уровня покрытия.

Достаточно много работ посвящено созданию подходов и методологий на основе машинного обучения, оптимизирующих процесс верификации. В диссертации [7] функциональная верификация была разделена на три этапа: планирование и создание тестов, их выполнение и обнаружение ошибок, а также разбор и классификация найденных ошибок. Для решения проблем в рамках этих трех этапов был предложен подход автоматизации, упорядочивания и приближения. В рамках данного подхода, на первом этапе определяется приоритетность определенных аспектов работы, затем автоматизируются действия, относящиеся к приоритетным аспектам, при этом используются приближения, которые снижают точность ради значительного выигрыша в эффективности.

Для эффективного планирования и создания тестов современных систем на кристалле, разработан автоматизированный процесс обнаружения высокоприоритетных аспектов верификации устройства. Кроме того, стало возможным создание более емких тестов, которые оказались до 11 раз более компактными, чем до применения данного процесса.

Для решения проблем в области выполнения тестов и обнаружения ошибок разработана группа решений, которые делают возможным внедрение автоматических и надежных механизмов для обнаружения недостатков устройства в процессе высокоскоростной функциональной верификации. Жертвуя точностью в пользу скорости, данные решения дают возможность выпускать платформы функциональной верификации, которые на более чем на три порядка быстрее традиционных платформ находят ошибки проектирования, которые в противном случае было бы невозможно обнаружить.

Проблемы в области диагностики ошибок решаются при помощи процесса, который полностью автоматизирует выслеживание дефектных компонентов устройства после обнаружения ошибки. Решение, которое обнаруживает дефектные устройства с более чем 70% точностью, значительно сокращает усилия по диагностике для каждой обнаруженной ошибки.

Ряд решений, использующих анализ данных, позволяет снижать трудозатраты на верификацию. В диссертации [8] предлагаются три методологии обучения на основе анализа данных для помощи инженерам-верификаторам в принятии одного из возможных решений (запустить больше тестов, усовершенствовать тестовый шаблон или же сменить его на новый) для достижения целей по функциональному покрытию.

Первая методология определяет важные тесты перед симуляцией, основываясь на том, что уже было промоделировано. Запуская только эти тесты и отбрасывая избыточные, можно сэкономить огромные ресурсы, такие как циклы симуляции. Вторая методология извлекает уникальные свойства из этих важных тестов, определенные при моделировании, и использует их для доработки тестового шаблона. Используя извлеченные сведения, создается больше тестов, схожих с важными. Созданные таким образом тесты более вероятно повышают уровень покрытия, которого было бы тяжелее достигнуть в ином случае. Третья методология анализирует набор существующих тестовых объектов (тестовых шаблонов) и определяет возможное дополнение плана тестирования. Автоматическое добавле-

ние новых тестовых объектов при помощи обучения на основе анализа данных, значительно снижает потребность в ручном труде инженеров-верификаторов по закрытию пробелов в тестовом плане покрытия.

Предлагаемые методологии обучения на основе анализа данных были разработаны и применены для верификации коммерческих микропроцессоров и устройств на платформе СнК. Результаты экспериментов демонстрируют осуществимость и эффективность построения систем анализа данных для повышения скорости и качества верификации.

Важным направлением является создание подходов, решающих проблему выбора тестов и минимизации тестовых наборов для достижения того или иного уровня покрытия. Особенно это актуально для регрессионного тестирования, так как позволяет экономить как время, так и вычислительные мощности. Одной из работ в этой области является диссертация [9], которая уделяет особое внимание использованию машинного обучения для снижения затрат времени на достижения целевых показателей покрытия, обычно занимающего наибольшую часть всего времени верификации. Для этой цели в двух различных экспериментах использовалось как глубокое обучение (Deep Q-Learning), так и обучение с подкреплением. С одной стороны, нейронные сети использовались для помощи с визуализацией, показывающей, стоит ли использовать тот или иной набор воздействий для моделирования, предсказывая уровень покрытия, который он создаст. С другой стороны, использовалось обучение с использованием функции полезности для предсказания минимального набора тестов, необходимого для достижения некоего уровня покрытия кода, путем оптимизации и уменьшения набора тестов, при котором будет достигаться все тот же уровень покрытия.

Результаты этих экспериментов показывают, что среднеквадратичная ошибка модели нейронной сети составляла от 3 до 5 единиц в предсказании двух различных величин покрытия соответственно, что является достаточно хорошим показателем для обучения на маленьком наборе данных. Во-вторых, Q-агент показал результаты на 43% лучше, чем утилита ранжирования покрытия инструмента моделирования (Questa RANKTEST). Применение данного метода может значительно уменьшить требуемое число моделирований в еженедельных регрессиях, что приведет к значительной экономии во времени и ресурсах.

Два вышеописанных подхода позволяют значительно сократить затрачиваемые командой инженеров-верификаторов ресурсы путем ускорения процесса верификации и его автоматизации.

В работе [10] предложена система Intelligent Test Selection (ITS), которая реализует онлайн-подход машинного обучения для обеспечения гибкого решения проблемы выбора тестов. ITS использует машинное обучение для оценки вероятности того, что тест обнаружит ошибку, вызванную конкретным изменением кода. Эти оценки используются для создания уменьшенного набора тестов, которые могут быть использованы для обнаружения скрытых ошибок по причине изменений кода с высокой степенью уверенности. Основные положения данной работы заключаются в следующем:

1) Детерминированный выбор тестов и способ определения приоритетов переопределены в вероятностные.

2) Единая схема для определения и отслеживания RTL-кода и тестов (направленных и случайных) в сложных компиляционных и функциональных потоках проекта.

3) Секционированная многоуровневая архитектура машинного обучения для оценки вероятности ошибки теста для данного изменения RTL-кода или тестов.

4) Динамический выбор функций и онлайн-метод обучения, который позволяет пользователям минимизировать затраты на сбор данных и поддерживать высокое качество результатов.

5) Эффективность подхода демонстрируется на пяти больших коммерческих устройствах и одном с открытым исходным кодом с различными размерами, сложностью и тестовыми методологиями.

Результаты экспериментов подтверждают полезность этого метода для различных стилей проектирования и тестирования. Данный метод с высокой точностью детектирует сбои и одновременно обеспечивает сокращение рекомендуемых наборов тестов для регрессий в 1,3 - 10 раз. Предлагаемый подход имеет ряд ограничений:

1) Сильная зависимость качества результатов от качества данных.

2) Низкий уровень качества результатов для ограниченно-случайных тестов.

Существуют работы, затрагивающие применение машинного обучения в более узконаправленных областях верификации. Одной из таких работ является статья [11], которая рассматривает область межуровневой верификации процессоров. В данной работе предлагается реализация случайного генератора, который на основании наблюдаемой информации о покрытии создает неограниченный поток инструкций, динамически развивающийся во время выполнения. В качестве эталонной модели в условиях тесной ко-симуляции используется ISS (эмулятор набора команд - исполняемая абстрактная мо-

дель процессорного ядра, как правило реализованная на C++). Информация о покрытии постоянно обновляется, основываясь на состоянии выполнения ISS. Подход применяет новую концепцию устаревания на основе покрытия для сглаживания распределения покрытия рандомизированного потока инструкций в течение времени. В сочетании это дает возможность для широкого и глубокого покрытия и обнаружения сложных ошибок в пограничных случаях (corner-case) в RTL-коде ядра.

Эксперименты с 32-битным конвейерным ядром RISC-V серии MINRES The Good Core (TGC) демонстрируют эффективность данного подхода. Было достигнуто намного более равномерное распределение покрытия случайного потока команд при помощи устаревания на основе покрытия, а также была обнаружена сложная микроархитектурная ошибка во взаимодействии уже тщательно протестированного индустриального процессора с прилагаемой инфраструктурой тестовой системы.

2.2. Использование машинного обучения в верификации с использованием UVM

UVM – стандартизированная методология, предназначенная для верификации цифровых устройств. UVM включает в себя библиотеку классов, что позволяет значительно ускорить процесс построения тестового окружения, а также переносить созданные компоненты окружения для верификации различных проектов.

Использование технологий машинного обучения в сочетании с UVM позволяет автоматизировать генерацию тестов и воздействий, что в теории позволяет значительно повысить скорость и качество верификации цифровых схем.

В исследовании [12] для этих целей использовалась трехслойная искусственная нейронная сеть. Входной слой был построен из 32 нейронов, скрытый и выходной – из 128. В качестве функции активации использовалась ReLU (Rectified Linear Unit). Целью внедрения являлось достижения требуемых показателей покрытия утверждений. В качестве DUT (Device under Test) служил простой ЦП. Использование нейронной сети для генерации позволило исключить до 40,2% ранее генерируемых стимулов. Время моделирования сократилось в 24,5 раза, а покрытие утверждений повысилось в 4-29 раз.

Недавние исследования в сфере машинного обучения часто фокусируются на «глубоком обучении» (*deep learning*). Подходы глубокого обучения делают сложные и очень точные выводы из массивных наборов данных. Глубокое обучение требует затратного по вычислительным мощностям процесса обучения и больших по сравнению с традиционным машинным обучением наборов данных, однако оно может обучаться высокоточным моделям, извлекать особенности и соответствия между данными автоматически и применять модели в разных приложениях. Исходя из возможностей глубокого обучения, в работе [13] было предложено использовать функции языка e в сочетании с UVM для обучения глубинной нейронной сети на основе результатов покрытия тестируемого процессора. Покрытие динамически считывалось с помощью Specman coverage API. Для обучения нейронной сети было запущено множество тестов в регрессионном режиме. Тестовое окружение поддерживает поэтапное выполнение, что позволяет в любой момент времени останавливать симуляцию, вводить новые собранные данные и повторно запускать ее. При использовании данного подхода время симуляции сократилось на 33%.

В работе [14] было произведено сравнение различных методов машинного обучения в рамках их применимости для генерации стимулов при верификации кэш-памяти четырехъядерного процессора. Особое внимание было уделено методам опорных векторов, глубинным нейронным сетям и ансамблю решающих деревьев. Для построения моделей машинного обучения использовались библиотеки Scikit-Learn и Keras языка Python. Тестовая среда была построена с помощью UVM. Результаты экспериментов показали, что машинное обучение может уменьшить суммарное время верификации на 70% даже с учетом времени обучения модели. Наиболее эффективным оказался ансамбль решающих деревьев — сокращение времени симуляции на 78%, в то время как метод опорных векторов и глубинные нейронные сети достигли сокращения на 69% и 77% соответственно. Предложенная методология обеспечивает полностью автоматический процесс, что позволяет избежать дорогостоящего ручного труда инженеров-верификаторов, требуемого при разработке направленных случайных тестов.

Также перспективными являются подходы, основанные на применении рекуррентных нейронных сетей (*RNN*), являющихся гибким инструментом, способным эффективно обрабатывать упорядоченные последовательности воздействий. Авторы статьи [15] предлагают новый подход к автоматизации верификации на основе покрытия при помощи оптимизатора на основе рекуррентной нейронной сети. Тестируемыми устройствами служили два процессора, Codasip uRISC и Codix Cobalt, заметно отличающиеся друг от друга уровнем сложности архитектуры. Для Codasip uRISC нейронная сеть состояла из

41 нейрона, для Codix Cobalt — из 1020. В результате применения данного подхода достижение 85% уровня покрытия потребовало примерно на 30% меньше времени, чем при классическом подходе без участия нейронных сетей или любого другого алгоритма оптимизации обработки обратной связи по покрытию.

Машинное обучение в сочетании с UVM также находит применение в области обнаружения и локализации ошибок. В работе [16] был разработан прототип инструмента, который применяет различные алгоритмы машинного обучения для кластеризации и классификации ошибок, обнаруженных в результате тестирования. В качестве входных данных использовались логфайлы симуляции тестовой системы UVM. Данные файлы были предварительно обработаны для создания функций, подходящих для алгоритмов машинного обучения.

Результаты экспериментов показали, что машинное обучение может быть эффективно для классификации первопричин неудачного прохождения тестов в системе UVM. Наиболее эффективным алгоритмом оказался метод «случайного леса» (random forest), обладающий точностью 0,907 и F1-мерой (среднее гармоническое значение точности и полноты) 0,913.

Применение машинного обучения для решения проблемы кластеризации неудачных тестов оказалось менее эффективным. Наибольшую эффективность продемонстрировал плотностный алгоритм кластеризации пространственных данных с присутствием шума в сочетании с методом главных компонент для уменьшения размерности. Показатель AMI (скорректированная взаимная информация) имеет значение 0,593, ARI (скорректированный индекс Рэнда) равен 0,545.

Данные результаты демонстрируют, что алгоритмы кластеризации могут быть недостаточно точными, чтобы на них полностью полагаться. Однако при использовании в сочетании с визуализацией алгоритмы кластеризации могут дать общее представление о том, какие неудачные прохождения тестов вызваны общей причиной.

3. Использование машинного обучения в «post-silicon» верификации

На сегодняшний день post-silicon верификация представляет собой в основном ручной, специализированный процесс. Начиная с первых прототипов микропроцессора, тестовые образцы подключаются к проверочной платформе, которая выполняет большие объемы тестов на высокой скорости. Результаты тестирования проверяются по эталонной модели или содержат самопроверку. До тех пор, пока результаты тестов совпадают, тестирование продолжается, в случае несовпадения результатов, выявляется сбой и начинается ручная отладка. Сначала необходимо воспроизвести сбой, что само по себе является проблемой для ошибок, чувствительных к малозаметным изменениям внутри кристалла. При возникновении таких ошибок, различные выполнения одного и того же теста дают разные результаты: одни проходят успешно, другие — нет. Зачастую труднее всего добиться именно неудачных выполнений.

Ввиду специфики данного этапа верификации и сложности процесса отладки, исследователи в этой области используют подходы машинного обучения для анализа возникших ошибок. К примеру, в статье [17] представлен подход к диагностике ошибок после изготовления кристалла, основанный на машинном обучении. Основанный на методе обнаружения аномалий алгоритм строит модель корректной активности сигнала на кристалле на основе прохождения тестов. Представленный алгоритм применяет методы обнаружения аномалий, схожие с используемыми для выявления мошенничества с кредитными картами, для обнаружения приблизительного цикла возникновения ошибки и набора возможных сигналов, являющихся причиной ошибки.

В сравнении с другими новейшими решениями в этой сфере, данный подход может определить время возникновения ошибки с примерно в 4 раза большей точностью при применении к сложному микропроцессору OpenSPARC T2.

Работа [18] предлагает использование технологий машинного обучения для автоматизации диагностики трассировочных данных устройства, а также для локализации ошибок в процессе посткремниевой верификации. Представленный набор инструментов позволяет создать алгоритм выбора сигналов, который определяет, какие сигналы отслеживать в процессе работы устройства. Выбор сигналов зависит от их типов, а также от связей между ними. При запуске тестируемого устройства трассировочные данные сохраняются для автономного анализа. Техника обработки больших данных, названная Map-Reduce, используется для преодоления проблем обработки огромного массива диагностических данных, полученных от запущенного на ПЛИС-прототипе устройства. Метод кластеризации k-средних применяется для группирования схожих сегментов данных диагностики и идентификации тех, которые встречаются редко во время работы устройства. Вдобавок, предложен набор инструментов для локализации ошибок, в котором кластеризация X-средних исполь-

зуется для группирования пройденных регрессионных тестов на кластеры так, что тесты с ошибками могут быть обнаружены, когда их не удается назначить ни одному их обученных кластеров. Экспериментальные результаты демонстрируют осуществимость предлагаемого подхода при устранении ошибок применительно к группе индустриальных устройств. Применение описанного метода позволяет обнаруживать дефекты устройств при помощи тестирования на основе мутаций.

4. Заключение

В данной статье был приведен краткий обзор использования методов машинного обучения при верификации цифровых устройств.

Наиболее широкое применение нашли данные методы в «pre-silicon» верификации, где машинное обучение в основном используется для генерации эффективных тестов и воздействий. Отдельного упоминания заслуживает использование машинного обучения в сочетании с UVM, которое позволяет достичь весьма значительного сокращения времени симуляции. В то же время достаточно мало работ было посвящено анализу данных на этапе «pre-silicon» верификации с целью автоматизации диагностики и локализации ошибок, а также оптимизации имеющихся тестовых наборов при сохранении уровня покрытия. Направление, посвященное анализу лог-файлов, является достаточно активно развивающимся в области тестирования программного обеспечения, наработки из данной сферы могли бы быть применены и для верификации микропроцессоров.

Машинное обучение также нашло применение в области «post-silicon» верификации, где оно используется для диагностики и локализации сбоев. Количество работ по данному направлению относительно невелико, что также делает его перспективной для исследования областью.

Проведенное авторами исследование применимости методов машинного обучения в верификации микропроцессоров и их моделей позволило определить направление дальнейших работ по внедрению этих инструментов в маршрут разработки современной элементной базы, принятый в ФГУ ФНЦ НИИСИ РАН. Наиболее перспективным направлением, согласно полученным в ходе работ результатом, является применение машинного обучения в сочетании с UVM. Также, эта тематика недостаточно раскрыта в работах российских ученых, следовательно, дальнейшие научные изыскания авторов будут направлены в эту область.

Публикация выполнена в рамках государственного задания ФГУ ФНЦ НИИСИ РАН по теме № FNEF-2022-0004.

A Survey of the Machine Learning Methods Applicability for Microprocessor Models Verification

A. D. Manerkin, N. A. Grevtsev, P. A. Chibisov

Abstract. The article provides an overview about the using machine learning methods in various areas of functional verification. We consider the use of machine learning in "pre-silicon" verification, precisely in simulation-based verification and Universal Verification Methodology. Then we discuss the field of "post-silicon" verification. The main decision was made in conclusion about the main areas of machine learning applications, as well as possible future directions of research.

Keywords: microprocessor verification, machine learning, Deep Learning, UVM

Литература

1. Камкин, А. С. Метод автоматизации имитационного тестирования микропроцессоров с конвейерной архитектурой на основе формальных спецификаций, диссертация на соискание ученой степени кандидата физико-математических наук; Федеральное государственное бюджетное учреждение науки Институт системного программирования им. В. П. Иванникова Российской академии наук. — Москва, 2009. — 181 с.

2. Hughes, William & Srinivasan, Sandeep & Suvarna, Rohit & Kulkarni, Maithilee. (2019). Optimizing Design Verification using Machine Learning: Doing better than Random.

3. K. H. Mami Miyamoto, Finding effective simulation patterns for coverage-driven verification using deep learning, in: SASIMI 2016 Proceedings, 2016, pp. 335–340.

4. A.S. Jagadeesh Accelerating coverage closure for hardware verification using machine learning, Master thesis, 2019, Texas A&M University

5. Danciu, Gabriel Mihail, and Alexandru Dinu. 2022. "Coverage Fulfillment Automation in Hardware Functional Verification Using Genetic Algorithms" Applied Sciences 12, no. 3: 1559. https://doi.org/10.3390/app12031559

6. S. L. Tweehuysen, G. L. A. Adriaans and M. Gomony, "Stimuli Generation for IC Design Verification using Reinforcement Learning with an Actor-Critic Model," 2023 IEEE European Test Symposium (ETS), Venezia, Italy, 2023, pp. 1-4, doi: 10.1109/ETS56758.2023.10174129.

7. Mammo, Biruk. (2017). Reining in the Functional Verification of Complex Processor Designs with Automation, Prioritization, and Approximation.

8. Chen, W. (2014). Data Learning Methodologies for Improving the Efficiency of Constrained Random Verification. UC Santa Barbara. ProQuest ID: Chen_ucsb_0035D_12213. Merritt ID: ark:/13030/m5sv2nkc. Retrieved from https://escholarship.org/uc/item/0db4j3jp

9. Aggoune M. (2022) Acceleration of Hardware Code Coverage Closure Using Machine Learning. University of Oulu, Degree Programme in Computer Science and Engineering, 56 p.

10. G. Parthasarathy et al., "RTL Regression Test Selection using Machine Learning," 2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC), Taipei, Taiwan, 2022, pp. 281-287, doi: 10.1109/ASP-DAC52403.2022.9712550.

11. N. Bruns, V. Herdt, E. Jentzsch and R. Drechsler, "Cross-Level Processor Verification via Endless Randomized Instruction Stream Generation with Coverage-guided Aging," 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE), Antwerp, Belgium, 2022, pp. 1123-1126, doi: 10.23919/DATE54114.2022.9774771.

12. Wang, F.; Zhu, H.; Popli, P.; Xiao, Y.; Bodgan, P.; Nazarian, S. Accelerating Coverage Directed Test Generation for Functional Verification: A Neural Network-Based Framework. In Proceedings of the Great Lakes Symposium on VLSI, ACM, New York, NY, USA, 23–25 May 2018; pp. 207–212.

13. Dinu, A.; Ogrutan, P.L. Opportunities of using artificial intelligence in hardware verification. In Proceedings of the 2019 IEEE 25th International Symposium for Design and Technology in Electronic Packaging (SIITME), Cluj-Napoca, Romania, 23 October 2019; pp. 224–227.

14. Gogri, S.; Hu, J.; Tyagi, A.; Quinn, M.; Ramachandran, S.; Batool, F.; Jagadeesh, A. Machine Learning-Guided Stimulus Generation for Functional Verification. In Proceedings of the Design and Verification Conference (DVCON-USA), Virtual Conference, 2–5 March 2020.

15. Fajcik, M.; Smrz, P.; Zachariasova, M. Automation of Processor Verification Using Recurrent Neural Networks. In Proceedings of the 2017 18th International Workshop on Microprocessor and SOC Test and Verification (MTV), Austin, TX, USA, 11–12 December 2017; pp. 15–20.

16. Truong, A.; Hellstrom, D.; Duque, H.; Viklund, L. Clustering and Classification of UVM Test Failures Using Machine Learning Techniques. In Proceedings of the Design and Verification Conference (DVCON), San Jose, CA, USA, 26 February–1 March 2018.

17. A. DeOrio, Q. Li, M. Burgess and V. Bertacco, "Machine learning-based anomaly detection for post-silicon bug diagnosis," 2013 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 2013, pp. 491-496, doi: 10.7873/DATE.2013.112.

18. Mandouh, Ema & Wassal, Amr. Application of Machine Learning Techniques in Post-Silicon Debugging and Bug Localization. Journal of Electronic Testing, 2018, 34. 1-19. 10.1007/s10836-018-5716-y.