Способ повышения качества алгоритмов выделения различий в текстовых файлах

С.В. Знаменский
E-mail: svz@latex.pereslavl.ru

Институт программных систем имени А.К. Айламазяна РАН

Национальный суперкомпьютерный форум, 24 ноября 2014 г.

Веськово, 2014

1. Задача выделения изменений в тексте
   - Общая постановка задачи
   - Потребности ручного анализа
   - Контекст в автоматической обработки
   - Родственные задачи теории и практики

2. Известные подходы и их тестирование
   - Основы известных подходов
   - Новый подход
   - Сравнение релевантности
   - Тесты

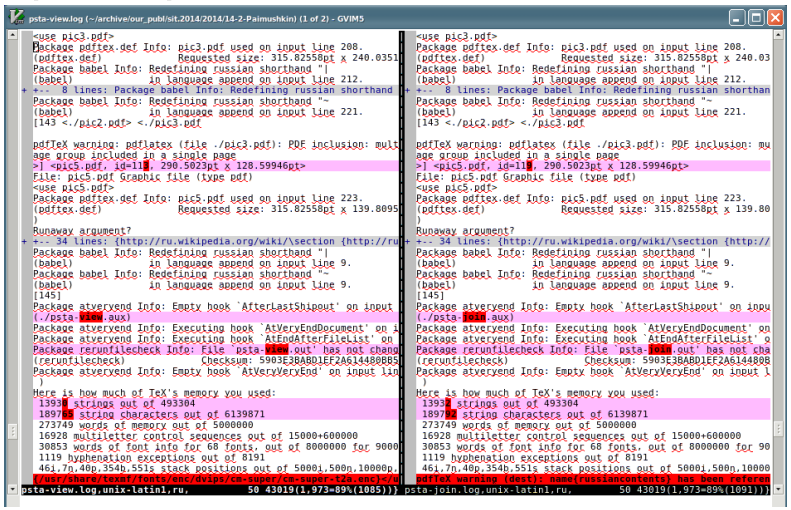
3. Вычислительная сложность

Проблема:

Найти алгоритм, лаконично, адекватно и за приемлемое время выделяющий различия между двумя версиями текста.

Анализ изменений в ходе обработки

Автоматически выявленные различия в логах обработки или трассировки системных вызовов



удобно использовать для диагностики проблем при отладке ПО.

Совместная работа над текстами

При совместной подготовке текстов полезны программы, обеспечивающие просмотр и редактирования различий между версиями текста.

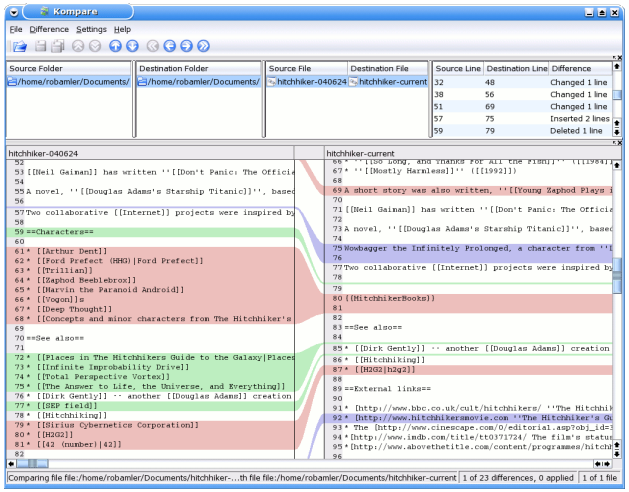
Beyond Compare	Scooter Software[1]	Proprietary	$US30 (std), 50 (pro)	1996	2014 (v4.0)
Compare++	Coode Software[2]	Proprietary	$29.95, 30-day free trial[3]	2010	2013 (v2.15)
diff, diff3	AT&T	BSD 3-clause, BSD 4-clause, CDDL, GPL, Proprietary	-	1974	
Ediff	Michael Kifer[5]	GPL	Free	1994	2.81.4
ExamDiff Pro	PrestoSoft[6]	Proprietary	$35, free file-only version[7]	1998	2014 (v7.0.1.6)
Far Manager (compare)	Eugene Roshal (original); FAR Group	Revised BSD license	Free	1996	August 2014 (v 3.0 build 4040)
fc	Microsoft[8]	Proprietary	Part of OS	1987	
FileMerge	Apple Inc.	Proprietary	Free (part of Apple Developer Tools)	1993 (part of NEXTSTEP 3.2[9])	2014 (v2.8)
Guiffy SureMerge	Guiffy Software[10]	Proprietary	$37.50(Pro), 75(eXpert)	2000	July, 2014 (v10.6)
IntelliJ IDEA (compare)	JetBrains[11]	Proprietary	Part of application		2007 (v6.0.5)
jEdit JDiff plugin	Various[12]	GPL	Free	1998	2012 (3.3.0)
Kompare	Otto Bruggeman[13]	GPLv2+	Free		
Lazarus Diff	Lazarus (software)	GPL	-	2000	2009
Meld	Stephen Kennedy[15]	GPLv2+	-	2002	2014-1-23 (v1.8.4)
Perforce P4Merge	Perforce	Proprietary[17]	-		2013 (2013.2)
Pretty Diff	Austin Cheney[18]	GPL-compatible	Free	2009	2012-07-05
Tkdiff	Tkdiff[19]	GPLv2+	Free	2003 (or before)	2011-11-27 (v4.2)
Total Commander (compare)	Christian Ghisler[20]	Proprietary	Part of application		2010-12-17 (v7.56a)
twdiff (TextWrangler Diff Helper)	Bare Bones Software, Inc.[21]	Proprietary	free (with TextWrangler [free])		2012 (1.0 (v22))
vimdiff	Bram Moolenaar et al.	GPL-compatible	-	2001	2006-05-07 (v7.0)
WinDiff	Microsoft[22]	Proprietary	Part of Platform SDK	1992	2003 ? (v6.1)
WinMerge	Dean Grimm[23]	GPL	Free	1998	2013-02-03 (v2.14.0)

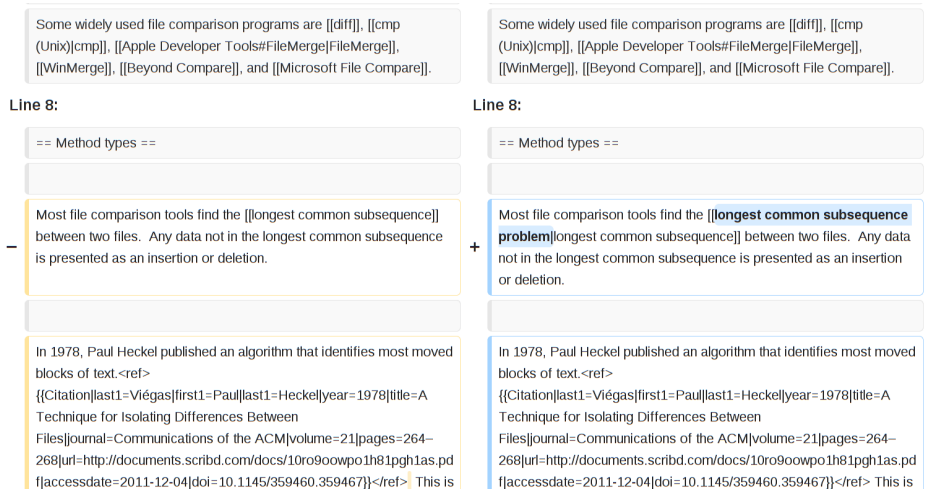
ЗАЧЕМ и КАКИЕ нужны суперкомпьютеры ~~экасафлопсного~~ эксафлопсного класса? Предсказательное моделирование свойств и многомасштабных процессов в материаловедении

Аннотация. Рассматривается подход, позволяющий выявить, для каких задач нужны суперкомпьютеры эксафлопсного класса. Возможности подхода рассмотрены на примерах актуальных задач материаловедения, физики конденсированного вещества и плотной плазмы, для решения которых необходимо атомистическое моделирование на современных и создаваемых в настоящее время суперкомпьютерах. Для каждой задачи проведено соответствие между набором изучаемых явлений и требуемым уровнем быстродействия (числа ядер) вычислительной системы. Показана масштабируемость параллельных программ моделирования и перспектива расширения предсказательной способности методов по мере увеличения числа вычислительных ядер и/или использования специализированных архитектур (графические ускорители). Рассмотрена иерархия методов моделирования, необходимых для адекватного описания свойств веществ на различных пространственных и временных масштабах. На наиболее глубоком нанометровом/пикометровом масштабе для моделирования электронной динамики и построения эффективных потенциалов взаимодействия частиц применяется теория функционала плотности (квантовая молекулярная динамика). Классический метод молекулярной динамики позволяет явно рассмотреть системы движущихся атомов вплоть до ~~микро-масштабов.~~ микромасштабов. Выход на ~~макро-масштабы~~ макромасштабы осуществляется с помо-

Элементы сравнения

Поскольку задача выделения изменений ресурсоёмкая, то тексты сравниваются построчно.





Чтобы длинные *различные* строки сравнивались по словам или символам (как в vim или в википедии), важно правильно их сопоставить.

Слияние версий исходных текстов

Контекст

При выделении и совместном применении одновременно внесённых разными авторами изменений эти изменения сохраняются с *контекстом*, в который обычно включаются три ближайших строки до и после каждого изменения.

Применение

При слиянии изменение применяется автоматически в том и только в том случае, если контекст остался неизменным после применения других изменений.

Слияние версий исходных текстов

Контекст

При выделении и совместном применении одновременно внесённых разными авторами изменений эти изменения сохраняются с *контекстом*, в который обычно включаются три ближайших строки до и после каждого изменения.

Применение

При слиянии изменение применяется автоматически в том и только в том случае, если контекст остался неизменным после применения других изменений.

Некоторые родственные задачи

- Поиск в словаре с учётом возможных опечаток,
- Синхронизация хранилищ данных,
- Расшифровка генома и родственные задачи биоинформатики,
- Антиплагиат.

Некоторые родственные задачи

- Поиск в словаре с учётом возможных опечаток,
- Синхронизация хранилищ данных,
- Расшифровка генома и родственные задачи биоинформатики,
- Антиплагиат.

Некоторые родственные задачи

- Поиск в словаре с учётом возможных опечаток,
- Синхронизация хранилищ данных,
- Расшифровка генома и родственные задачи биоинформатики,
- Антиплагиат.

Некоторые родственные задачи

- Поиск в словаре с учётом возможных опечаток,
- Синхронизация хранилищ данных,
- Расшифровка генома и родственные задачи биоинформатики,
- Антиплагиат.

Некоторые родственные задачи

- Поиск в словаре с учётом возможных опечаток,
- Синхронизация хранилищ данных,
- Расшифровка генома и родственные задачи биоинформатики,
- Антиплагиат.

1. Количество вставок, замен и удалений

Метрика Левенштейна

- Широко используется с начала 80-х.
- Рекуррентная формула, динамический алгоритм сложности $O(N \times M)$.

2. Длина общей подпоследовательности

Longest Common Subsequence (LCS)

- Основа diff, GNU diffutils, perl и многих других программ.
- Длина LCS просто выражается через количества вставок, удалений и замен.
- Алгоритм Майера: Сложность $O(N \times M)$, для ограниченных изменений $O(N \times D)$.

3. LCS с акцентом на уникальные элементы

bzrlib/patiencediff.py

- Алгоритм Кохена: выделяются уникальные элементы и строится CS для них, дальше рекурсия.
- Работает быстрее, хотя оценки сложности хуже.

4. Присутствие длиннейших совпадающих строк

Bdiff (difflib)

- Алгоритм Ратклифа-Мезнера: ищется длиннейшая общая подстрока, затем рекурсия.
- Быстр. Широко используется в python, Mercurial, Revlog.

5. Длины цепочек примыкающих вставок/удалений

Модели биоинформатики

- Алгоритм Смита и Ватермана, Нидлмана — Вунша: кроме количества вставок, замен и удалений учитывает длины цепочек соседних вставок или удалений.
- Суммарный штраф за длины цепочек минимизируется.
- Используются различные зависимости размеров штрафов от длин цепочек.

Идея: минимизировать вероятность конфликтов при слиянии

Критерий минимальной вероятности конфликтов

Количество общих элементов, не попавших в контекст, должно быть максимально

Иными словами

Суммарное превышение длин общих сегментов над удвоенным размером контекста должно быть максимально

Обозначим P_k суммарное превышение длин общих сегментов над числом k.

Зависимость от размера контекста

Критерии для различных k

$k = 0$ Длина P_0 подпоследовательности CS ($\Longrightarrow$ LCS).

$k = 1$ Количество P_1 смежных пар в CS

$k = 2$ Количество P_2 внутренних (не примыкающих к изменённым) элементов CS

$k > 2$ ...

Критерии различны и каждый имеет слабости.

Зависимость от размера контекста

Критерии для различных k

$k = 0$ Длина P_0 подпоследовательности CS ($\Longrightarrow$ LCS).

$k = 1$ Количество P_1 смежных пар в CS

$k = 2$ Количество P_2 внутренних (не примыкающих к изменённым) элементов CS

$k > 2$ ...

Критерии различны и каждый имеет слабости.

Зависимость от размера контекста

Критерии для различных k

$k = 0$ Длина P_0 подпоследовательности CS ($\Longrightarrow$ LCS).

$k = 1$ Количество P_1 смежных пар в CS

$k = 2$ Количество P_2 внутренних (не примыкающих к изменённым) элементов CS

$k > 2$ ...

Критерии различны и каждый имеет слабости.

Зависимость от размера контекста

Критерии для различных k

$k = 0$ Длина P_0 подпоследовательности CS ($\Longrightarrow$ LCS).

$k = 1$ Количество P_1 смежных пар в CS

$k = 2$ Количество P_2 внутренних (не примыкающих к изменённым) элементов CS

$k > 2$ ...

Критерии различны и каждый имеет слабости.

Зависимость от размера контекста

Критерии для различных k

$k = 0$ Длина P_0 подпоследовательности CS ($\Longrightarrow$ LCS).

$k = 1$ Количество P_1 смежных пар в CS

$k = 2$ Количество P_2 внутренних (не примыкающих к изменённым) элементов CS

$k > 2$ ...

Критерии различны и каждый имеет слабости.

Идея: сложить все P_k

Необычный простой критерий

S_q =Сумма квадратов длин отрезков CS

Система оценивания тестов релевантности моделей

Семантика отметок релевантности по тесту

- $+$ правильный вариант однозначно предпочтителен
- $-$ неправильный вариант однозначно предпочтителен
- $\pm$ любой ответ на конкретный тест можно гарантированно получить задавая подходящие параметры
- ? Результат непредсказуем: критерии близости, на которых основана модель, не могут различить хороший и плохой вариант.

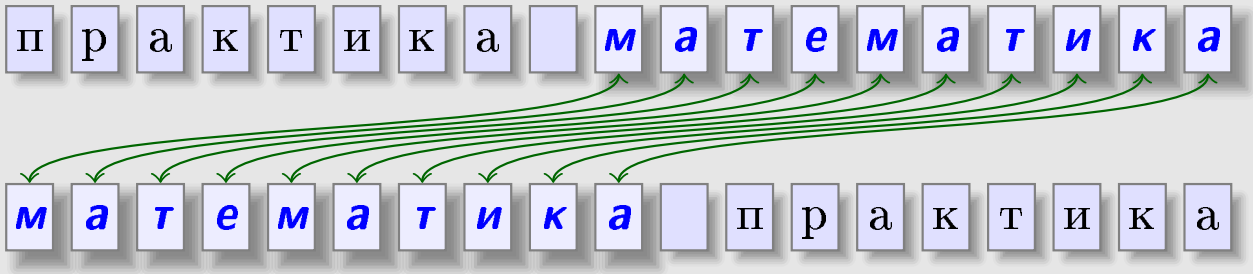
Итоги тестирования релевантности

Таблица отметок релевантности

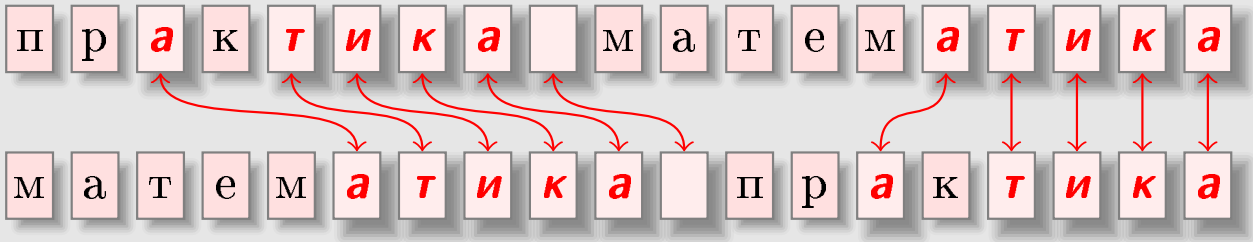
№	Ключевой критерий модели	Номер теста					
		1	2	3	4	5	6
1	Количество вставок, замен и удалений	−	−	+	+	+	?
2	Длина P_0 общей подпоследовательности (CS)	−	−	+	+	+	?
3	Длина CS из уникальный элементов	−	+	+	+	+	?
4	Наличие длиннейших совпадающих строк	+	+	−	−	−	+
5	Количество вставок, замен и удалений с учётом длин цепочек изменений	±	±	±	±	±	−
6	Количество P_1 соседей в CS	+	+	−	+	+	+
7	Количество P_2 внутренних совпадений	+	+	?	−	+	+
8	Сумма квадратов длин отрезков CS	+	+	+	+	+	+

Тест 1

лучше



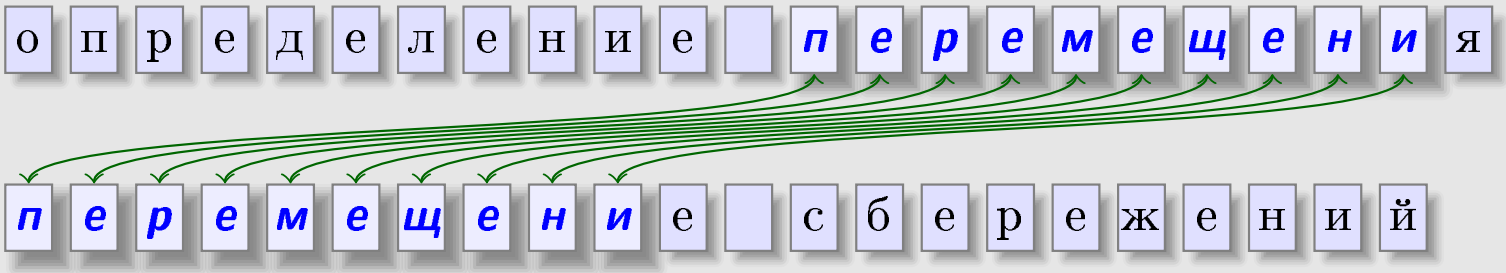
хуже



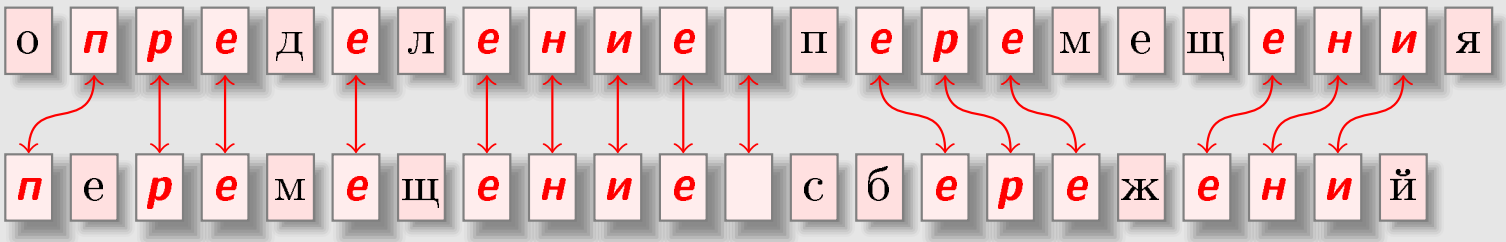
Разумеется, 11 разрозненных случайных совпадений букв существенно менее значимы, чем 10 подряд идущих.

Тест 2

лучше



хуже

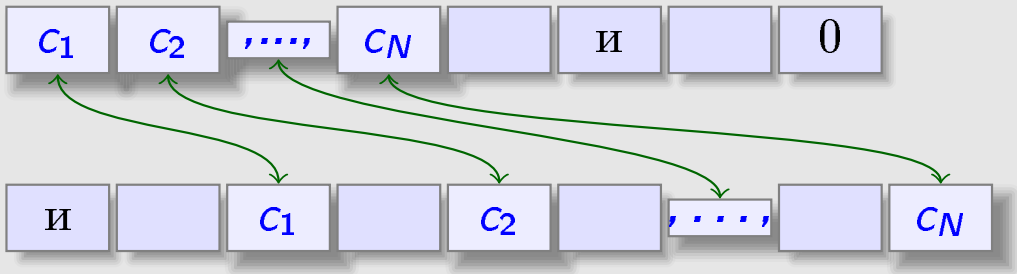


При сравнении исходных файлов программ совпадение 10 подряд идущих строк надёжнее свидетельствует о непосредственной связи фрагментов, чем совпадение прореженных 15 строк (которые могут содержать к примеру по одной скобке).

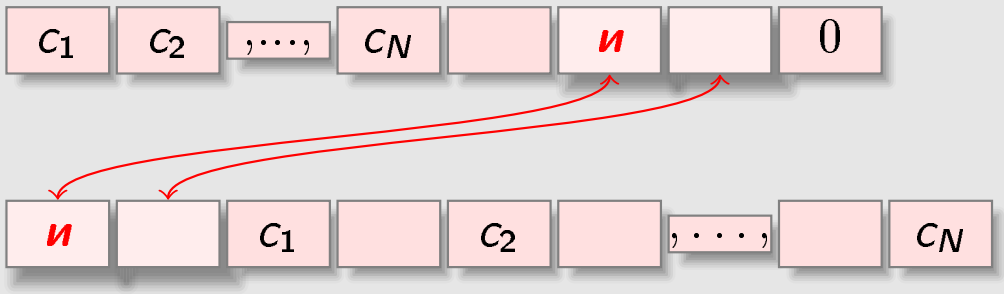
Тест 3

Если N много больше мощности множества, которому принадлежат члены последовательностей, то

лучше



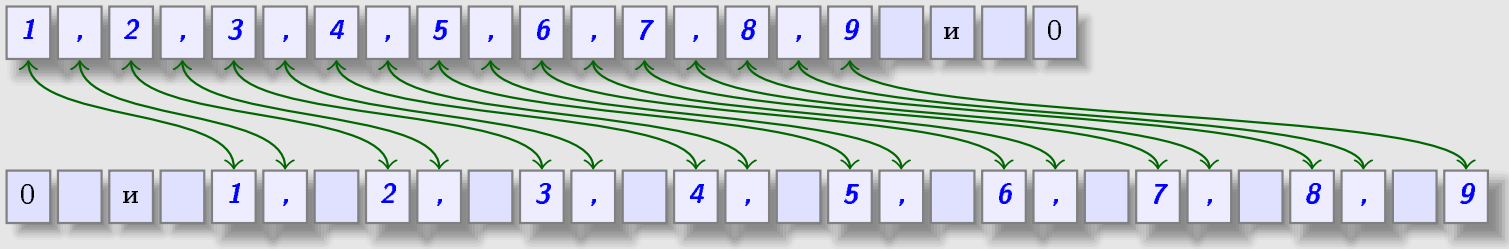
хуже



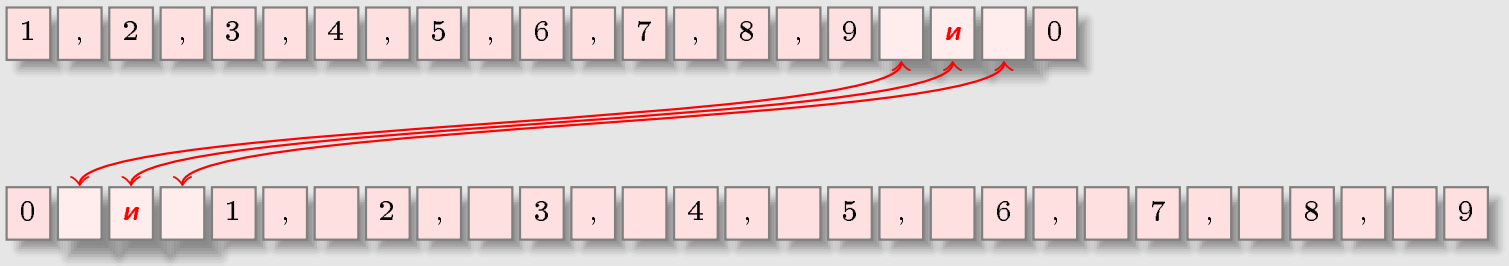
где ,...., изображает $c_3, \ldots, c_{N-1}$, а ,...., их же «в разрядку».

Тест 4

лучше



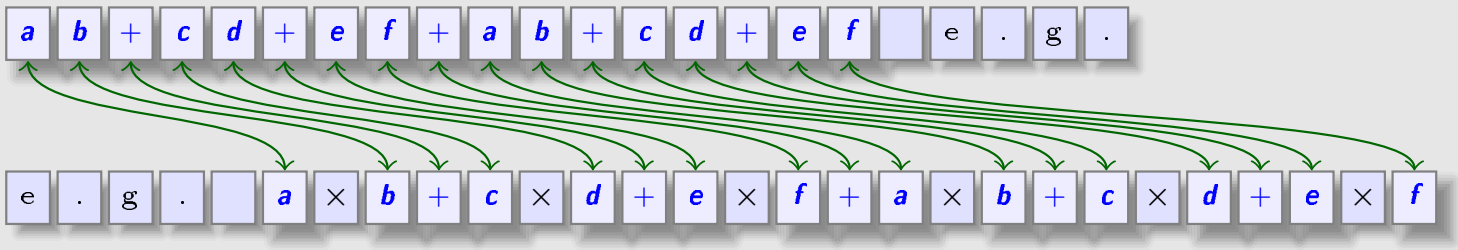
хуже



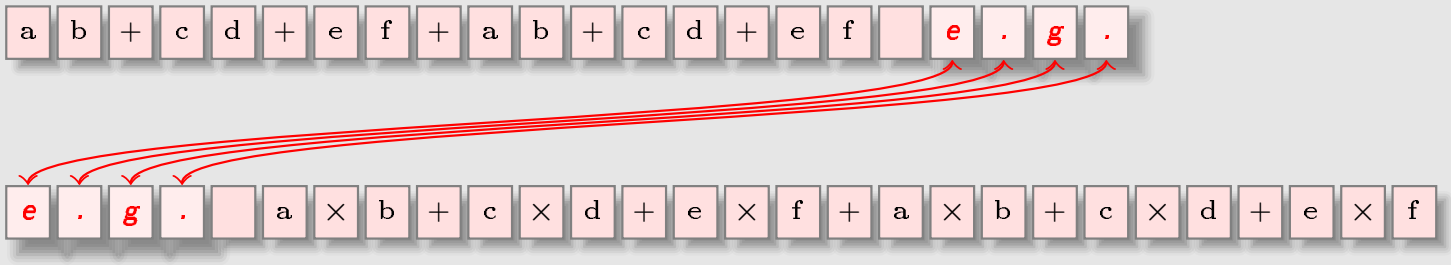
Легко модифицировать пример, увеличив последовательных парных совпадений до заведомо более значимого, чем одно тройное.

Тест 5

лучше



хуже

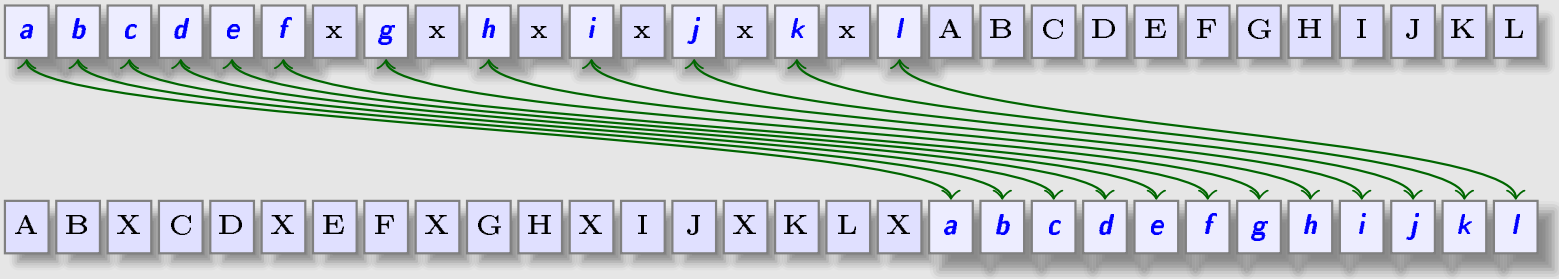


Совпавший в нижнем случае фрагмент

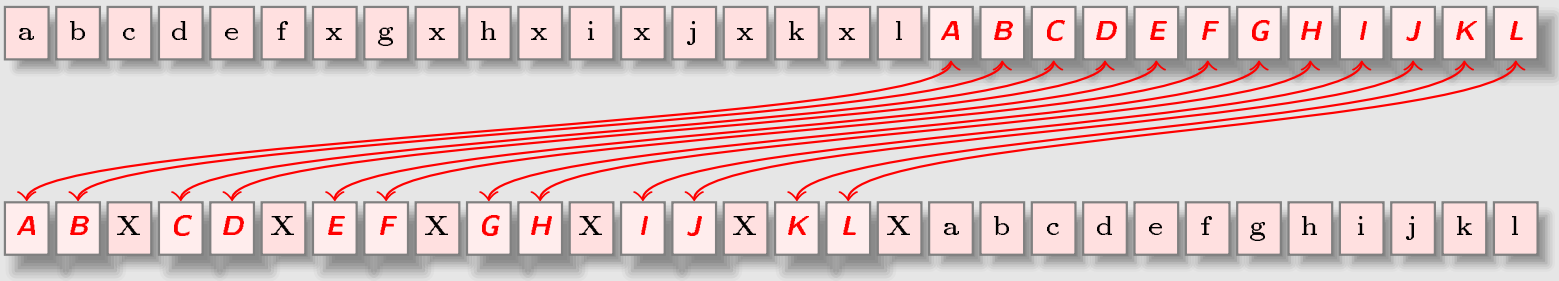
менее значим, чем цепочка из большого достаточно большого числа совпадающих отрезков на единицу меньшей длины.

Тест 6

лучше



хуже



Верхний вариант патча оставляет возможность бесконфликтных независимых изменений подстроки a b c d, а нижний бесполезен, поскольку конфликтует с любым другим патчем.

Новый подход выглядит более релевантным

Идеи алгоритмов и оценки сложности

будут представлены в докладе