

Федеральное агентство по образованию
Государственное образовательное учреждение
высшего профессионального образования
Петрозаводский государственный университет
Математический факультет

Кафедра информатики
и математического обеспечения

ПРОЕКТ АРХИТЕКТУРЫ
для расширения программной системы
WEB-SYNDIC

Заказчик:
к. ф.-м. н.,
доцент Корзун Д. Ж.

Последняя модификация:
Срд Май 2 00:57:34 MSD 2007

Рабочая версия

Петрозаводск — 2007

Содержание

1	Общее описание	3
2	Архитектура Web-SynDic	3
2.1	Server	3
2.2	Client Part	4
2.3	Data Store	5
3	Изменения вносимые в систему Web-SynDic	5
4	Архитектура Algorithm Server, описание изменений	5
5	Структуры данных	6
5.1	Класс TrafficAnalyzerParameters	6
5.2	Класс CFGrammar	7
5.3	Класс NetworkGraph	7
5.4	Класс GrammarBuilderParams	8
5.5	Класс GrammarToANLDEConverterParams	8
5.6	Класс GraphGeneratorParameters	8
5.7	Класс GeneralGeneratorSpooler	9
5.8	Класс GeneralGeneratorTask	10
6	Интерфейс пользователя	10
6.1	Обработка графов	12
6.2	Справка	13
6.3	Обработка неверного формата данных	14
6.4	Обработка потоков	14

1 Общее описание

Разрабатываемая система представляет собой модификацию Web-SynDic. Основная задача при проектировании архитектуры — обеспечить дальнейшую интеграцию в существующую архитектуру.

2 Архитектура Web-SynDic

Web-SynDic делится на три подсистемы: «Server» (координирует действия между модулями), «Client Part» (часть системы с которой непосредственно работает пользователь) и «Data Store»(осуществляет обработку и хранение данных о пользователях).

Также, есть внешние модули, которые осуществляют генерацию и поиск решения систем одАНЛДУ. Схема архитектуры приведена на рис. 1

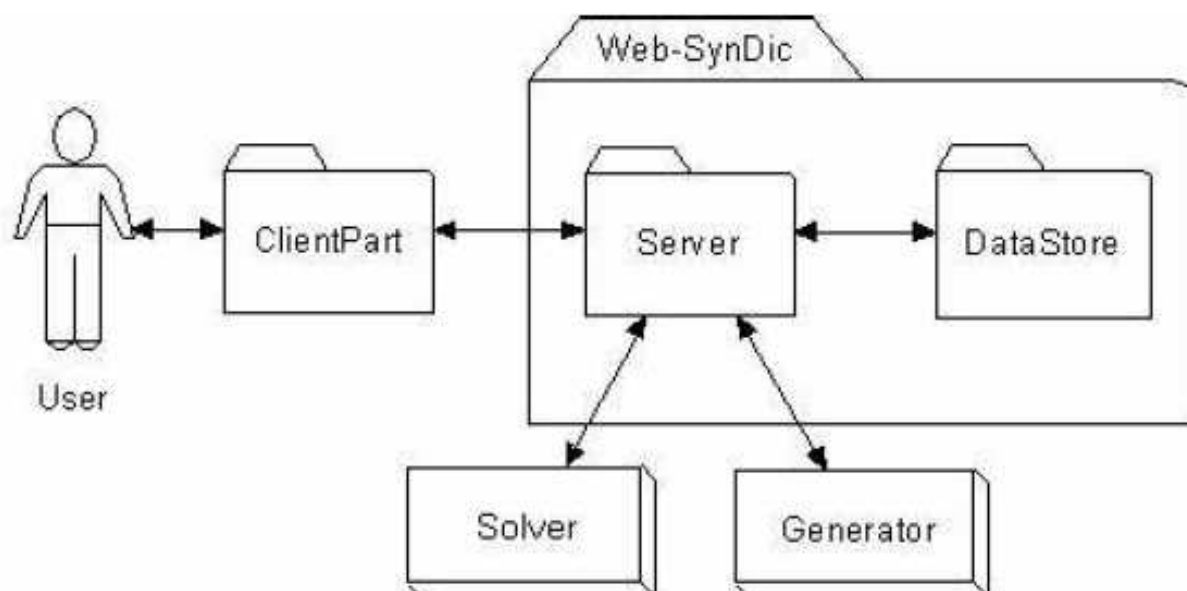


Рис. 1: Высокоуровневая архитектура WEB-SYNDic

2.1 Server

Подсистема «Server» состоит из пяти модулей (рис. 2).

Модуль «Web Server» представляет интерфейс между пользователем и системой «Web-SynDic». Web Server получает запросы и параметры от client part, преобразует их во внутренний формат пересылает далее для обработки другим модулям.

Модуль «Session Processing» управляет всеми запущенными сессиями, координирует передачу пользовательского потока данных между модулями, подсистемами.

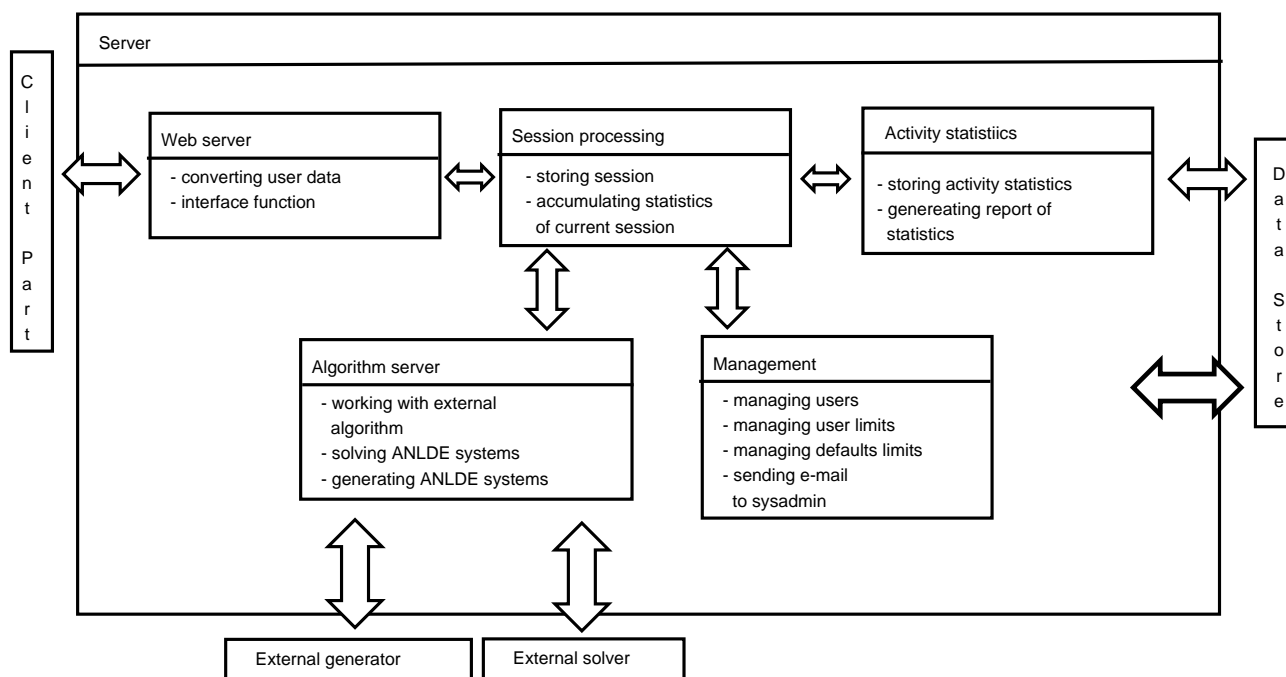


Рис. 2: Модуль «Server»

Модуль «Algorithm Server» осуществляет работу с внешними алгоритмами. Т.е. запускает алгоритм всякий раз, когда это требуется пользователем в течение сессии и передает результаты выполнения модулю «Session Processing».

Модуль «Management» управляет профилями пользователей и ограничениями, используемыми по умолчанию. Все важные данные сохраняются модулем «Data store». Зарегистрированный пользователь может изменять только свой профиль. Администратор может менять любой профиль, а также ограничения.

Модуль «Activity statistics» контролирует и обновляет деятельность каждого пользователя, сохраненную модулем «Data Store» и вычисляет статистику для администратора системы.

2.2 Client Part

Данный модуль является интерфейсной частью между пользователем (ПО пользователя) и подсистемой «Web Server». Через данный модуль пользователь может получить доступ ко всем функциям системы, которые ему разрешено использовать.

Функции, которые выполняет данный модуль можно разделить на пять групп:

- 1) Функции отвечающие за работу с АНЛДУ системами.
- 2) Функции уведомления.
- 3) Функции работы с пользователем, получения статистики, а также функции для управления ограничениями.

- 4) Функции осуществляющие регистрацию и «вход» пользователя в систему.
- 5) Функции для работы с сессиями.

2.3 Data Store

Модуль «Data Store» предназначен для хранения данных о разегистрированных пользователях, статистической информации (активности пользователя) и параметров, которые используются при генерации/решении АНЛДУ.

3 Изменения вносимые в систему Web-SynDic

Модификация системы представляет собой добавление определенных модулей, для осуществления генерации систем уравнений АНЛДУ на основе анализа потоков данных и топологии сети. Т.е. главные изменения коснутся подсистемы «Algorithm Server».

4 Архитектура Algorithm Server, описание изменений

Структура классов «Algorithm Server» показана на рис. 3.

Работа данного модуля состоит в том, чтобы генерировать и решать системы АНЛДУ.

Классы GenTask и SolTask, которые являются потомками класса Task, содержат данные и параметры для генерации/решения. Все задачи хранятся в соответствующих спулерах (SolverSpooler и GeneratorSpooler). В спулерах имеется буфер для хранения задач, также они хранят список всех доступных для работы решателей, генераторов. Когда подходит очередь на выполнение очередной задачи, осуществляется вызов класса, который выполняет работу с нужным генератором. После окончания процесса генерации или решения, задача удаляется из буфера, также вызывается метод "solved" или "generated" который посылает сигнал системе, после этого система может работать с результатами решения или со сгенерированными системами АНЛДУ. Как было сказано выше, существуют классы, которые осуществляют работу с внешними алгоритмами, с внешними модулями. Каждый такой класс является потомком классов «Solver» или «Generator». Для подключения нового внешнего алгоритма к системе, необходимо описать класс, который будет осуществлять всю работу по взаимодействию с подключаемым модулем.

Т.о. для того, чтобы расширить систему предполагается добавить классы на основе класса Task, также необходимо подключить спулер, который будет работать с задачами по генерации систем АНЛДУ на основе анализа трафика и анализа топологии р2р-сети.

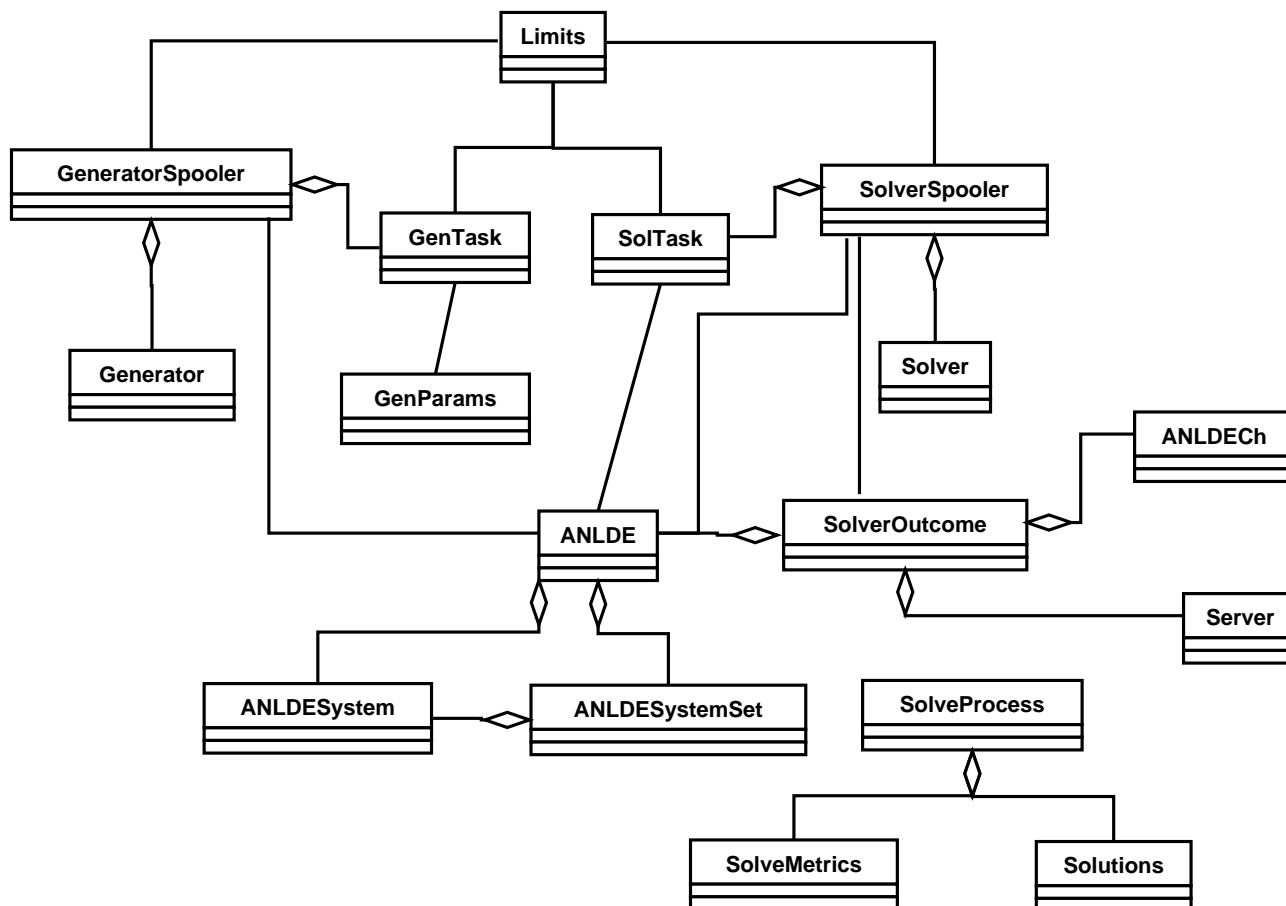


Рис. 3: Архитектура Algorithm Server

5 Структуры данных

5.1 Класс TrafficAnalyzerParameters

Служит для хранения параметров алгоритмов.

```

public class TrafficAnalyzerParameters
{
    int windowSize; // хранит размер окна алгоритма генерации разбиения.

    List fileNames; // хранит список имен файлов NetFlow, которые
    подлежат исследованию.

    int fileCount; // хранит количество обрабатываемых файлов.

    int partitionAlgorithmID; // хранит идентификатор алгоритма генерации
    разбиения.

    int partitionPrecision; // точность округления в алгоритме генерации
    разбиения.
}
  
```

```
int passagePrecision; // точность определения состояний в алгоритме
построения переходов.
```

```
int passageBorder; // минимальное количество появлений одного и того
же перехода для того, чтобы он учитывался при построении системы АНЛДУ.
Те, что встречаются реже, отбрасываются.
}
```

5.2 Класс CFGGrammar

Класс описывает контекстно-свободную грамматику

```
public class CFGGrammar
{
    private int[] _leftParts; // Левые части грамматики
    private int[][] _rightParts; // Правые части грамматики

    // Создает новый экземпляр грамматики
    public CFGGrammar( int[] leftParts, int[][] rightParts );

    // Создает новый экземпляр грамматики из строкового описания
    public CFGGrammar( String strGrammar, String partsSeparator );

    // Преобразует грамматику в строковую интерпретацию
    public String toString( String partsSeparator );
}
```

5.3 Класс NetworkGraph

Служит для хранения графа сгенерированной р2р-системы.

```
public class NetworkGraph
{
    private int[] _nodes; // Вершины графа
    private int[][] _links; // Связи между вершинами

    // Создает новый экземпляр графа
    public NetworkGraph( int[] nodes, int[][] links );

    // Возвращает массив вершин графа
    public int[] getNodes();

    // Возвращает массив связей между вершинами
```

```

    public int[][] getLinks();
}

```

5.4 Класс GrammarBuilderParams

Служит для хранения данных, которые применяются при построении КС-грамматики по графу.

```

public class GrammarBuilderParams
{
    private NetworkGraph _graph; // Хранит граф

    // Создает новый экземпляр класса
    public GrammarBuilderParams( NetworkGraph netGraph );

    // Возвращает граф для обработки
    public NetworkGraph getNetworkGraph();
}

```

5.5 Класс GrammarToANLDEConverterParams

Служит для хранения данных, которые применяются при построении КС-грамматики по графу.

```

public class GrammarToANLDEConverterParams
{
    private CFGGrammar _grammar; // Хранит описание контекстно-свободной
    грамматики

    // Создает новый экземпляр класса
    public GrammarToANLDEConverterParams( NetworkGraph netGraph );

    // Возвращает грамматику
    public CFGGrammar getGrammar();
}

```

5.6 Класс GraphGeneratorParameters

Служит для хранения параметров генератора .

```

public class GraphGeneratorParameters
{
    private int _nodeCount; // Хранит количество вершин.
}

```



```

// Создает пустой объект.
public GraphGeneratorParameters();

// Создает объект, при этом устанавливает количество вершин.
public GraphGeneratorParameters(int nodeCount);

// Устанавливает количество вершин
public int setNodeCount( int value );

// Возвращает количество вершин
public int getNodeCount();
}

```

5.7 Класс GeneralGeneratorSpooler

Общий спулер для задач генерации.

```

public class GeneralGeneratorSpooler
{
    private List _buffer; // Используется для создания очереди задач.
    private List _workerList; // Хранит список обработчиков.

    private String _name; // Хранит имя спулера.
    private String _profilePath; // Хранит путь к файлу настроек.

    private int _bufferSize; // Хранит размер буфера.

    private Thread _thread;
    // Для работы с нитью
    public GeneralGeneratorSpooler( String profilePath, int bufferSize,
Server ch );

    // Возвращает количество задач, которые в данный момент находятся в
спулере. public int getTasks();

    // Возвращает список имен обработчиков.
    public String[] getWorkerList();

    // Возвращает название спулера.
    public String getName();

    // Обработка потока
    public void start();
}

```

```
// Запуск потока
public void stop();

// Останавливает поток
public void run();

}
```

5.8 Класс GeneralGeneratorTask

Класс для хранения данных о задаче.

```
public class GeneralGeneratorTask extends Task
{
    private GeneratorConfiguration _configuration; // Ограничения на
    выполнение обработки.
    private String _generatorName; // Имя используемого генератора.
    private object _parameters; // Параметры для обработчика.

    // Создает экземпляр класса с заданными параметрами.
    public GeneralGeneratorTask( ANLDE anlde, GeneratorConfiguration
    configuration, string generator, object parameters );

    // Устанавливает имя используемого генератора
    public setGeneratorName( String Name );

    // Возвращает ограничения
    public GeneratorConfiguration getConfiguration();

    // Возвращает имя генератора
    public String getGeneratorName();

    // Возвращет параметры для генератора
    public object getParameters();

}
```

6 Интерфейс пользователя

Все ниже описанные страницы интерфейса необходимо реализовать с помощью технологии jsp, добавив к уже существующим необходимые страницы.

Компоненты меню:

- 1) Главное меню.

2) Меню пользователя.

3) Статус пользователя.

Главное меню

Основные функции:

- Генерация систем АНЛДУ;
- Генерация набора систем АНЛДУ;
- Генерация графа;
- Работа с потоками.

По сравнению с текущей версией интерфейса Web-SynDic к основным функциям системы добавились функции генерации графа и обработки потоков.

Документация:

- Теория решения систем АНЛДУ;
- Теория по генерации графов;
- Теория работы с потоками.

По сравнению с текущей версией интерфейса Web-SynDic к документам по теории решения систем АНЛДУ добавились документы по теории генерации графов и теории работы с потоками.

Меню пользователя:

- Ссылка на связь с администратором системы для отправки замечаний и пожеланий по улучшению системы;
- Конфигурация алгоритмов — ссылка на страницу с ограничениями по работе с алгоритмами.

Статус пользователя:

Назначение элементов:

Текстовая область «Nickname» предназначена для ввода имени пользователя.

Текстовая область «Password» предназначена для ввода пароля.

Кнопка «Log In» предназначена для входа в систему зарегистрированного пользователя.

Кнопка «Register» предназначена для регистрации нового пользователя.

Process a graph

Input or generate a graph

[Get help with graph format](#)

Generate automatically a graph which corresponds to topology of the [Chord](#) infrastructure of specified size

Generator parameters	Current value	Max. value
Number of nodes:	<input type="text" value="8"/>	15

Store this graph in text format (please, allow browser to show popup windows)

After you have entered topology, you can generate ANLDE system based on it. When the system is generated, you will be redirected to the page where it can be solved.

Рис. 4: Страница обработки графа

6.1 Обработка графов

Пользователю предоставляется возможность задать или сгенерировать граф топологии сети. Для этих целей в интерфейс системы добавляется страница, изображенная на рис. 4.

Назначение элементов интерфейса:

Текстовая область «Graph» предназначена для ввода графа пользователем или отображения сгенерированного графа в виде списка ребер (см.¹ раздел Форматы данных).

¹ *Технический комментарий:* Несуществующий пока

Кнопка «**Generate**» вызывает генератор графа. После генерации граф отображается в поле **Graph**.

Кнопка «**Save**» сохраняет граф в текстовый файл.

Кнопка «**Generate ANLDE System**» осуществляет генерацию системы АНЛ-ДУ на основе графа, описываемого в поле **Graph**

6.2 Справка

Также пользователю предоставляется возможность получить справку о верном формате графа. Для этой цели в интерфейс системы добавляется страница, изображенная на рис. 5.

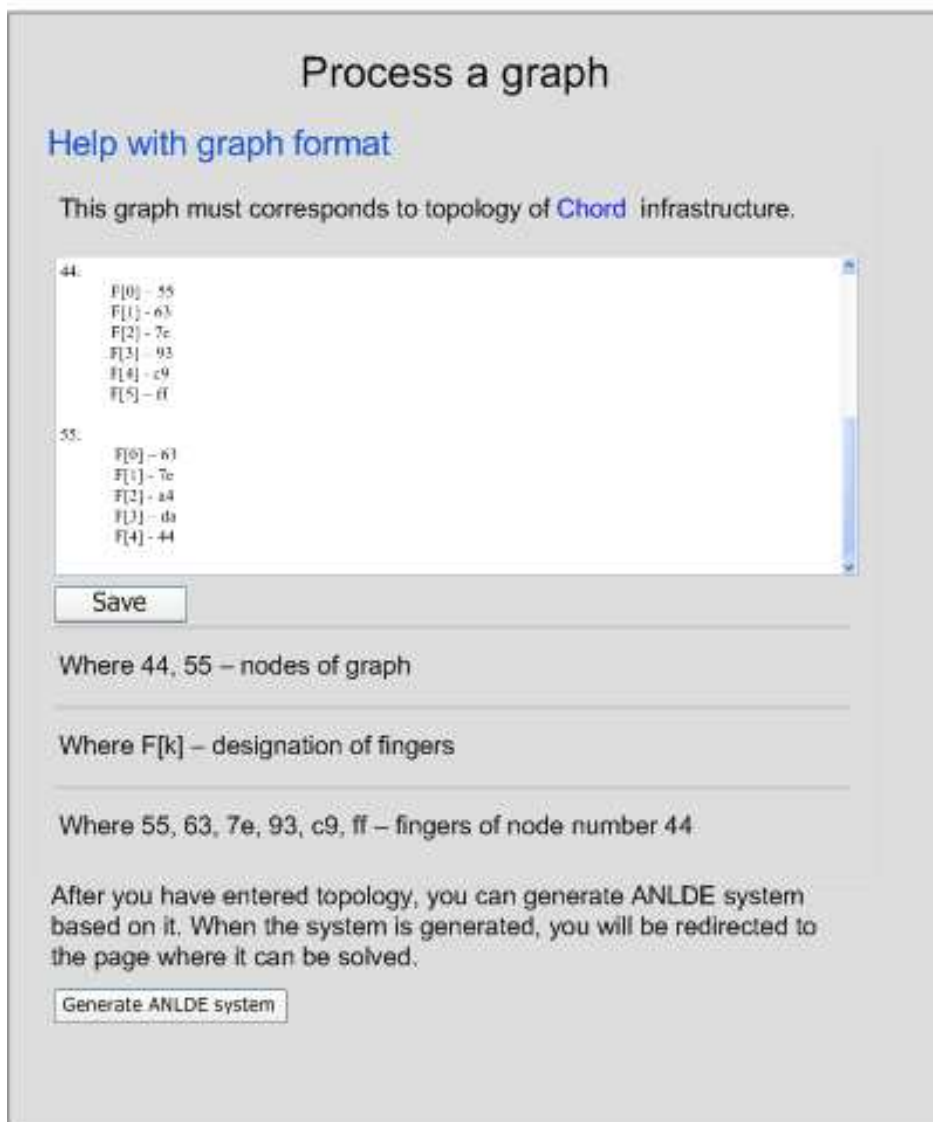


Рис. 5: Страница примера правильного формата графа

Назначение элементов интерфейса:

Текстовая область «Graph» предназначена для отображения примера графа в виде списка ребер.

Объяснение назначения каждой строчки описанного примера.

Кнопка «Generate» вызывает генератор графа. После генерации граф отображается в поле **Graph**.

Кнопка «Save» сохраняет граф в текстовый файл.

Кнопка «Generate ANLDE System» осуществляет генерацию системы АНЛ-ДУ на основе графа, описываемого в поле **Graph**

6.3 Обработка неверного формата данных

При условии неверно введенного формата графа, пользователь увидит страницу, изображенную на рис. 6.

Данная страница отличается от страницы стандартной обработки графов (рис. 4) тем, что пользователь видит, что он совершил ошибку при вводе формата графа и ему настоятельно предлагается получить справку о верном формате вводимых данных.

Назначение элементов элементов интерфейса:

Сообщение «ERROR» предназначено для оповещения пользователя о неверном формате входных данных.

Ссылка «Get help with graph format» предназначена для перехода на страницу с верным форматом входных данных.

Текстовая область «Graph» предназначена для отображения примера графа в виде списка ребер.

Кнопка «Generate» вызывает генератор графа. После генерации граф отображается в поле **Graph**.

Кнопка «Save» сохраняет граф в текстовый файл.

Кнопка «Generate ANLDE System» осуществляет генерацию системы АНЛ-ДУ на основе графа, описываемого в поле **Graph**

6.4 Обработка потоков

Пользователю предоставляется возможность работы с потоками. Для этого в интерфейс системы добавляется страница, изображенная на рис. 7.

Назначение элементов интерфейса:

Process a graph

ERROR:

Wrong format of data.

[Get help with graph format](#)

Input or generate a graph

[Get help with graph format](#)

Dgvfgfdgb
 Gnnbnbnm

Generate automatically a graph which corresponds to topology of the [Chord](#) infrastructure of specified size

Generator parameters	Current value	Max. value
Number of nodes:	<input type="text" value="8"/>	15

Store this graph in text format (please, allow browser to show popup windows)

After you have entered topology, you can generate ANLDE system based on it. When the system is generated, you will be redirected to the page where it can be solved.

Рис. 6: Страница обработки неверного формата графа

Текстовая область «Transitions system» предназначена для ввода системы переходов пользователем или отображения системы переходов после ее генерации на основе потоков.

Кнопка «Save» сохраняет систему переходов в текстовый файл.

Выпадающий список «Files» позволяет выбрать файл с потоками в формате NetFlow.

Кнопка «Generate Transitions system» позволяет сгенерировать систему переходов, которая удовлетворяет выбранному файлу с потоками по соответствующему алгоритму разбиения.

Кнопка «Generate ANLDE system» позволяет сгенерировать систему АНЛДУ по системе переходов.

Process a NetFlows

Input or generate Transitions system

Store this transitions system in text format (Please, allow browser to show popup windows)

Choose alloting algorithm.

Choose NetFlow file

Parameters	Current value	Max. value
Window size of alloting algorithm:	<input type="text" value="3"/>	100
Alloting precision:	<input type="text" value="6"/>	150
States' defining precision:	<input type="text" value="10"/>	1000
Low border of sagnificance (transitions alg):	<input type="text" value="100"/>	10000

Рис. 7: Страница обработки потоков