Підручник з Rocs

Tomaz Canabrava
Andreas Cord-Landwehr
Переклад українською: Юрій Чорноіван

Підручник з Rocs

Зміст

Анотація

Rocs — інструмент побудови та перегляду графів.

Розділ 1

Вступ

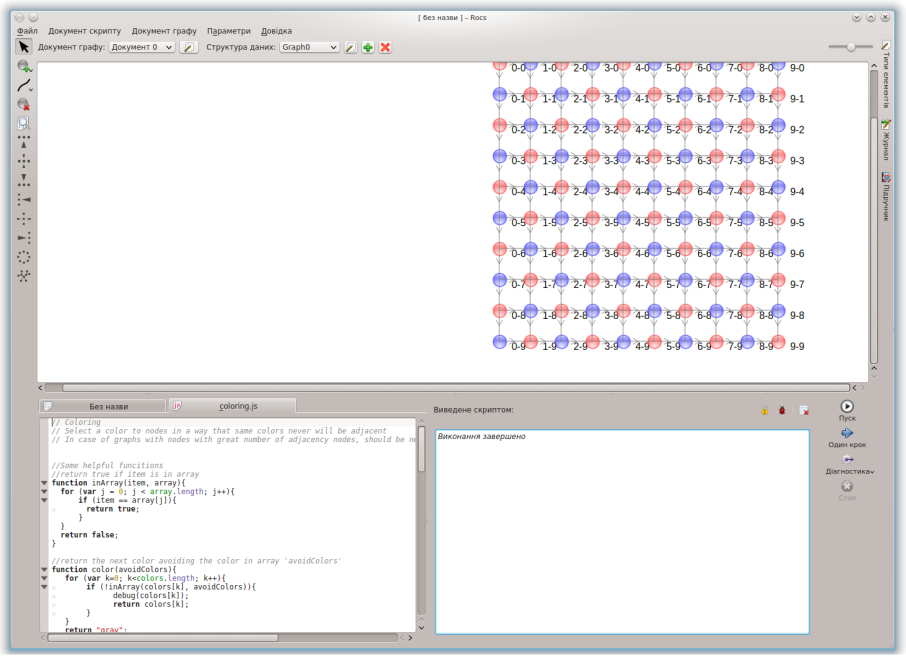
У цьому розділі наведено короткий огляд основних можливостей та прийомів роботи у Rocs. Користувачам, які хочуть негайно перейти до навчання принципам роботи у Rocs, ми пропонуємо перейти до читання розділу Розділ 1.2, користуючись настановами розділу розділ 3 щодо написання алгоритмів.

1.1 Мета, цільова аудиторія програми та процес роботи з нею

Rocs — комплексний інструмент для теорії графів, призначене для всіх, хто цікавиться розробкою та вивченням алгоритмів, пов'язаних з графами. Це зокрема

- викладачі, які хочуть продемонструвати роботу алгоритмів, пов'язаних з графами, студентам,
- студенти і дослідники, які бажають краще розібратися у роботі алгоритмів,
- усі, хто цікавиться структурами даних та алгоритмами.

Для усіх цих користувачів у Rocs передбачено простий у користуванні графічний редактор для створення графів, потужний рушій для роботи зі скриптами та виконання алгоритмів, а також декілька допоміжних інструментів для імітацій, експериментів та експортування графів. Типовим способом використання Rocs є створення графу або вручну (тобто перетягуванням вершин і ребер на полотно), або а допомогою засобів створення графів. Після цього можна реалізувати і виконати алгоритми обробки графів для створеного графу. Усі внесені алгоритмом зміни буде негайно показано у редакторі графів.



1.2 Ядро Rocs

Загалом, кожен сеанс Rocs є проєктом: під час запуску Rocs створюється порожній проєкт; якщо завантажуєтеся якийсь проєкт, він стає поточним. Сам проєкт складається з *документів графів*, *скриптів або алгоритмів* та *журналу*.

1.2.1 Документи графів

Документ графу відтворює вміст полотна у редакторі графів. У ньому містяться дані щодо визначених користувачем типів вузлів та ребер, їхніх властивостей, а також щодо вже створених вузлів і ребер. Тобто, Rocs вважає набір усіх вузлів і ребер документа графів (необов'язково з'єднаних) самим графом. Доступ до усіх елементів документа графу можна отримати за допомогою рушія обробки скриптів та загального об'єкта `Document`.

1.2.2 Типи ребер

Іноді, граф може складатися із ребер різних типів (наприклад, незорієнтований граф плюс ребра дерева, обчислені за допомогою алгоритму пошуку у ширину), які слід обробляти і показувати у різний спосіб. Для цього, окрім типового типу ребер, ви можете визначити довільні інші типи ребер. Кожен тип ребер має власне візуальне представлення, динамічні властивості, а також може бути орієнтованим або незорієнтованим. У інтерфейсі написання скриптів передбачено зручні методи для доступу до ребер певних типів.

1.2.3 Типи вузлів

Аналогічно до типів ребер ви можете визначити різні типи вершити графу (наприклад, надати певним вершинам особливі ролі). У кожного типу вершин (вузлів) передбачено власне представлення та динамічні властивості.

1.2.4 Властивості

Кожен елемент (вершини або ребра) може мати властивості. Ці властивості має бути визначено у відповідному типі вершин або ребер. Властивості ідентифікуються за назвами і можуть містити будь-які значення. Доступ до властивостей виконується за назвами. Для створення або внесення змін до наявних властивостей слід використовувати бічну панель **Типи елементів**. Для відкриття діалогового вікна властивостей слід використовувати кнопку **Властивості**.

Крім того, ви можете скористатися рушієм обробки скриптів для доступу до зареєстрованих властивостей та зміни їхніх значень. У наведеному нижче прикладі ми припускаємо, що властивість «weight» зареєстровано для типового типу ребер.

```
var nodes = Document.nodes()
for (var i = 0; i < nodes.length; ++i){
    nodes[i].weight = i;
}
for (var i = 0; i < nodes.length; ++i){
    Console.log("weight of node " + i + ": " + nodes[i].weight);
}
```

1.3 Покрокові настанови

У цьому розділі ми створимо приклад проєкту для вивчення декількох найважливіших можливостей Rocs. Метою є створення графу та скрипту для ілюстрування простого 2-апроксимаційного алгоритму розв'язування задачі *мінімального вершинного покриття*. Задача мінімального вершинного покриття полягає у визначенні підмножини вузлів графу C мінімального розміру, такої, що кожне з ребер графу з'єднано з принаймні одним вузлом з C. Відомо, що ця задача має NP-складність. Ми хочемо проілюструвати спосіб визначення парної апроксимації обчисленням відповідника вказаного графу.

Нашою метою буде візуалізації зв'язку між відповідностями і мінімальним вершинним покриттям. Для цього ми визначимо два типи ребер: один для показу відповідних ребер і ще один для показу звичайних ребер; а також два типи вузлів, за допомогою яких відрізнятимемо вузли, що належать до C, і вузли, що не належать до C.

1.3.1 Створення графу

Для створення графу скористаймося типовим засобом створення графів Rocs. Викликати цей засіб можна за допомогою пункту основного меню **Документ графу** → **Інструменти** → **Створити граф**. Виберемо **Випадковий граф** з 30 вузлами, 90 ребрами і зреном 1 (зерном є початкове значення для генератора випадкових значень; використання одного зерна призведе до створення тих самих графів).

1.3.2 Створення типів елементів

Ми скористаємося панеллю **Типи елементів** і створимо другий тип вершин, а також другий тип ребер. Для обох нових типів ми відкриємо діалогове вікно властивостей за допомогою відповідних кнопок **Властивості** і встановимо ідентифікатори у значення `2`. Далі, ми змінимо кольори елементів цих двох нових типів (для того, щоб їх було просто відрізнити від типових типів). Нарешті, ми встановимо двоспрямований тип ребер і визначимо для ідентифікаторів типових типів значення `1`.

1.3.3 Алгоритм

Нарешті, нам слід реалізувати алгоритм апроксимації. Для цього ми скористаємося таким кодом:

```
for (var i=0; i < Document.nodes.length; i++) {
    Document.nodes[i].type = 1;
}
for (var i=0; i < Document.edges.length; i++) {
    Document.edges[i].type = 1;
}

var E = Document.edges(); // набір необроблених ребер
var C = new Array();      // відповідні ребра
while (E.length
> 0) {
    var e = E[0];         // беремо перше ребро e={u,v}
    var u = e.from();
    var v = e.to();
    e.type = 2;           // робимо ребро відповідним
    E.shift();            // вилучити e (тобто E[0]) зі списку ребер
    C.push(u);            // додати u до C
    C.push(v);            // додати v до C

    // позначити u,v як вузли у C
    u.type = 2;
    v.type = 2;

    // вилучити з E усі ребра, суміжні з u або v
    var adjacent = u.edges();
    for (var i=0; i < adjacent.length; i++) {
        var index = E.indexOf(adjacent[i]); // знайти індекс
        if (index != -1) {
            E.splice(index, 1); // вилучити його, якщо знайдено
        }
    }
    var adjacent = v.edges();
    for (var i=0; i < adjacent.length; i++) {
        var index = E.indexOf(adjacent[i]); // знайти індекс
        if (index != -1) {
            E.splice(index, 1); // вилучити його, якщо знайдено
        }
    }
}
Console.log("Покриття вершин містить " + C.length + " вузлів.");
```

1.3.4 Виконання алгоритму

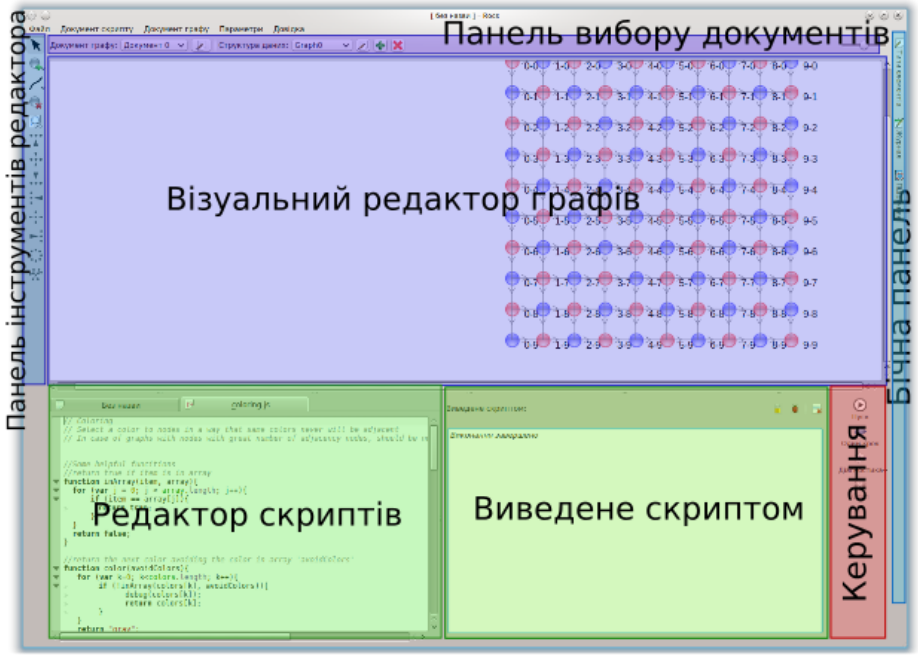
Алгоритм можна виконати за допомогою кнопки  **Виконати** на панелі керування скриптами.

Розділ 2

Інтерфейс користувача Rocs

2.1 Основні елементи інтерфейсу користувача

Інтерфейс користувача програми поділено на декілька логічних частин, як це показано на наведеному нижче знімку.



Редактор графів

У редакторі передбачено панель полотна, на якій можна розташовувати вершини і ребра. Подвійне клацання на будь-якому з елементів цієї панелі відкриває відповідне меню властивостей. Ви можете скористатися цими інструментами за допомогою *вкладок бічної панелі* для створення та редагування графів.

Доступні інструменти:

- Згори ліворуч на панелі розташовано вказані нижче кнопки дій. Натискання кнопки дії, означатиме, що вказівник миші виконуватиме відповідну дію на панелі редактора графів:

–  **Позначити і пересунути**: щоб позначити елементи, натисніть ліву кнопку миші на порожньому місці на графі, утримуйте кнопку натисненою і рухом вказівника миші створіть прямокутник, у якому міститимуться елементи даних та/або ребра. Якщо треба позначити окремий елемент, просто клацніть лівою кнопкою миші на цьому елементі. Якщо ви наведете вказівник миші на позначений елемент або набір елементів, натиснете і утримуватимете натисненою ліву кнопку миші, а потім почнете рухати вказівник, разом з ним почнуть рухатися позначені елементи. Пересунути позначені елементи можна також за допомогою клавіш зі стрілочками.

–  **Створити вузол**: клацніть у довільній точці панелі редактора, щоб створити новий вузол, який належатиме поточній позначеній структурі даних. Якщо ви продовжите утримувати кнопку миші, програма покаже контекстне меню, за допомогою якого ви зможете вибрати тип даних нового елемента (якщо передбачено різні типи даних елементів).

–  **Створити ребро**: наведіть вказівник миші на позицію першого вузла, натисніть і утримуйте натисненою ліву кнопку миші, перетягніть вказівник до позиції другого вузла, до якого має прямувати ребро. Виконати цю дію можна буде, лише якщо у поточному графі можливе додавання такого ребра (наприклад, у незорієнтованому графі не можна додавати декілька ребер між двома вузлами). Якщо ви утримуватимете натисненою кнопку миші і далі, програма покаже меню, за допомогою якого можна буде вибрати тип ребра (якщо типи ребер передбачено поточним типом графу).

–  **Вилучити елемент**: клацніть на елементі графу, щоб його вилучити. Якщо ви вилучите вузол, всі ребра, які з ним з'єднано, також буде вилучено.

Бічна панель

Праворуч розташовано бічну панель, за допомогою якої можна отримати доступ до декількох робочих інструментів:

- **Типи елементів**: за допомогою цього віджета можна отримати безпосередній доступ до доступних типів ребер і вузлів.
- **Журнал**: у кожного з проєктів є власний журнал, яким можна скористатися, наприклад, для занотовування завдань, результатів та спостережень.
- **Програмний інтерфейс для роботи зі скриптами**: за допомогою цього віджета можна отримати безпосередній доступ до документації зі скриптів.

Редактор скриптів

У полі цього текстового редактора ви можете створювати скрипти алгоритмів у спосіб, описаний у розділі розділ 3. Ви можете працювати одночасно над декількома алгоритмами: для цього достатньо відкрити або створити декілька вкладок скриптів за допомогою головного меню програми.

Дані, виведені скриптом:

До цього поля програма виводитиме діагностичні повідомлення або дані, виведені скриптом, залежно від перемикача у верхній частині цього віджета. Якщо під час виконання скрипту станеться помилка, програма негайно перемкнеться у режим діагностичних повідомлень.

Керування

На цій панелі розташовано елементи керування виконанням скриптів. Ви можете наказати програмі виконати скрипт, відкритий у редакторів скриптів, натисканням кнопки  **Пуск**. Якщо виконання скрипту ще не завершено, його можна зупинити натисканням кнопки **Стоп**.

Розділ 3

Скрипти

3.1 Виконання алгоритмів у Rocs

Rocs внутрішньо використовує рушій Java Script QtScript. Це означає, що усі реалізовані вами алгоритми має бути написано мовою Java Script. Нижче ми розберемо те, як отримати доступ та змінити елементи документа графу за допомогою рушія роботи зі скриптами. Важливо зауважити, що зміни, внесені за допомогою рушія обробки скриптів, безпосередньо буде відображено у властивостях елементів редактора графів.

3.1.1 Керування виконанням скриптів

Передбачено різні режими виконання ваших алгоритмів.

-  **Виконати**: виконати скрипт, аж до завершення його роботи.
-  **Стоп**: припинити виконання скрипту (доступне, лише якщо виконується скрипт).

3.1.2 Дані, виведені скриптом

Під час виконання алгоритму діагностичні повідомлення та повідомлення, виведені програмою, буде показано на панелі *Вивід повідомлень*. Якщо рушієм обробки скриптів буде виявлено синтаксичну помилку у вашому скрипті, відповідне повідомлення також буде показано серед діагностичних повідомлень. Зауважте, що всі повідомлення, виведені програмою, також буде показано і у списку діагностичних повідомлень (напівжирним шрифтом).

Керувати текстом, який буде виведено скриптом, можна за допомогою таких функцій:

```
Console.log(string message);            // показати повідомлення як виве ↩
    дене скриптом
Console.debug(string message);          // показати повідомлення як діаг ↩
    ностичне
Console.error(string message);          // показати повідомлення як поми ↩
    лку
```

3.1.3 Інтерфейс (API) роботи зі скриптами

Кожна з різних частин Rocs надає статичний елемент, доступ до якого можна отримати за допомогою рушія обробки скриптів. Цими частинами є:

- `Document` — документ графу;
- `Console` — виведені до консолі записи журналу.

Явне використання програмного інтерфейсу та довідку з методів можна знайти у вбудованій довідці бічної панелі Rocs.

Розділ 4

Імпортування та експортування

4.1 Обмін проєктами Rocs

Проєкти Rocs можна імпортувати та експортувати у форматі файлів архівів `.tar.gz`. Такими архівами можна скористатися для обміну проєктами, імпортування та експортування можна здійснювати за допомогою пунктів меню **Документ графу** → **Імпортувати проєкт...** та **Документ графу** → **Експортувати проєкт як**, відповідно.

4.1.1 Імпортування та експортування документів графів

У поточній версії Rocs передбачено підтримку імпортування та експортування даних файлів у таких форматах:

- файли DOT, також відомі як файли Graphviz
- файли GML
- Файли у форматі TGF
- Формат KML

4.1.1.1 Формат файлів звичайних графів (TGF)

Звичайний формат графів (Trivial Graph Format або TGF) є простим текстовим форматом файлів, призначеним для опису графів. Файл TGF містить список визначень вузлів з прив'язкою ідентифікаторів вузлів до міток, за яким вказано список ребер. У такому форматі можливий запис лише однієї мітки на вузол і одного значення на ребро. Rocs обробляє імпортовані графи у цьому форматі як неорієнтовані. Експортовані графи у цьому форматі міститимуть по два ребра на з'єднання, якщо з'єднання у них є двонапрямленими.

4.1.1.1.1 Специфікація формату

- Файл починається зі списку вузлів (один вузол на рядок), далі має бути рядок з єдиним символом «#», за яким має йти список ребер (по одному ребру на рядок).
- Запис вузла складається з цілого числа (ідентифікатора), пробілу і довільного рядка.
- Запис ребра складається з двох цілих чисел (ідентифікаторів), відокремлених пробілом, пробілу та довільного рядка. Вважається, що спрямоване ребро виходить з вузла з першим ідентифікатором і прямує до вузла з другим ідентифікатором.

4.1.1.1.2 Приклад

```
1 starting node
2 transmitter
3 sink
#
1 2 blue
2 1 red
2 3 green
```

4.1.1.2 Мова DOT та формат файлів графів Graphviz

Мова DOT є мовою текстового опису графів, у якій поєднується придатне до читання людиною представлення графів та можливості ефективної обробки графу програмами компонування. DOT є типовим форматом файлів для комплексу програм для візуалізації графів Graphviz, але цей формат також широко використовується іншими програмами для обробки графів. Типовими суфіксами назв файлів у форматі DOT є `.gv` та `.dot`.

4.1.1.2.1 Непідтримувані можливості

Програма Rocs здатна обробляти всі файли графів, які містять дані, що відповідають специфікації мови DOT[1]. Передбачено повну підтримку можливостей мови, окрім таких виключень:

- підграфи: через відсутність концепції підграфів у Rocs, підграфи (subgraph) імпортуються лише як набори елементів даних зі з'єднаннями. Зокрема, не імпортуються дані щодо вхідних і вихідних з'єднань між підграфами.
- Атрибути HTML і XML: атрибути (зокрема мітки) що містять синтаксичні конструкції HTML або XML читаються без обробки. Зокрема, не виконується обробка інструкцій щодо зміни шрифтів та стилю запису з атрибутів.

4.1.1.2.2 Приклад

```
digraph myGraph {
    a -> b -> c;
    b -> d;
}
```

[1] https://graphviz.org/doc/info/lang.html

Розділ 5

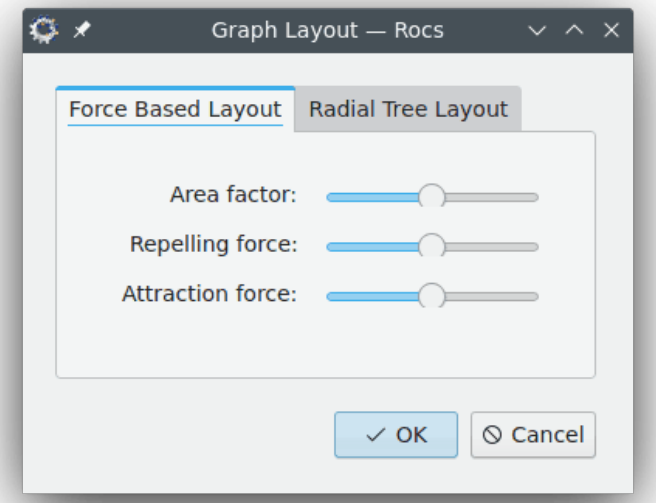
Компонування графу

5.1 Автоматичне компонування графів у Rocs

Rocs може виконувати компонування графів автоматично. Доступ до засобу компонування графів Rocs можна отримати з головного меню програми: **Документ графу** → **Інструменти** → **Компонування графу**. Для компонування можна застосовувати два різних алгоритми: компонування на основі сил і компонування з радіальним деревом. Щоб застосувати одне з компонувань, виберіть відповідну вкладку вікна засобу компонування графів, вкажіть бажані параметри і виконайте алгоритм натисканням кнопки **Гаразд**. Подробиці щодо кожного з алгоритмів компонування наведено у нижче.

5.1.1 Компонування на основі сил

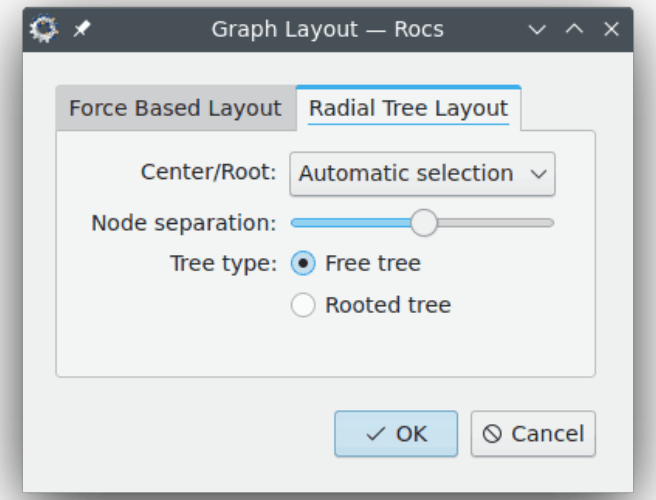
Компонування на основі сил може бути застосоване до будь-яких графів. Якщо описувати цей алгоритм просто, він імітує сили, які діють на кожен з вузлів графу. Між сусідніми парами вузлів діють сили відштовхування і притягання. Величини цих сил можна вказати пересуванням відповідних повзунків у інтерфейсі користувача.



Іншим параметром, який можна змінювати є коефіцієнт площі. Цей параметр керує розсіюванням вузлів за площею. У компонуваннях, які створено із більшими значеннями коефіцієнта площі, відстань між вузлами є більшою.

5.1.1.1 Компонування з радіальним деревом

Компонування «радіальне дерево» може бути застосовано лише до дерев. Будь-які спроби застосувати цей алгоритм компонування до інших типів графів призводитимуть до повідомлення про помилку. Параметри компонування «радіальне дерево» може бути вибрати за допомогою відповідного інтерфейсу користувача.



Параметр типу дерева надає змогу вибрати між довільним деревоподібним компонуванням і кореневим деревоподібним компонуванням. У довільному компонуванні вузли розташовуються довільно без видимої ієрархії. У кореневому компонуванні кореневий вузол розташовується згори, а усі підлеглі дерева — під ним; це компонування надає змогу бачити ієрархію вузлів.

Параметр центру або кореня визначає, який з вузлів буде використано як кореневий для кореневого компонування або як центр для довільного компонування. Центром довільного компонування є перший вузол, який оброблятиметься алгоритмом. Усі інші вузли буде розташовано на колах із центром у центральному вузлі. Центр або корінь може бути вибрано алгоритмом компонування у автоматичному режимі.

Параметр відокремлення вузлів керує відстанню між вузлами. Збільшення значення цього параметра спричиняє збільшення відстані між вузлами. Відповідно, зменшення значення робить відстань між вузлами меншою.

Розділ 6

Подяки і ліцензія

Rocs

Переклад українською: Юрій Чорноіван yurchor@ukr.net