Силовые алгоритмы
(Л. 6)

Апанович З.В.
apanovich@iis.nsk.su
Тел:3309344
К. 217

Методы размещения, основанные на физических аналогиях

- Методы, основанные на физических аналогиях очень популярны по следующим причинам:
- 1) они очень интуитивны;
- 2) их легко понять и запрограммировать;
- 3) для графов размером порядка 150 вершин дают вполне удовлетворительные результаты;
- 4) размещения, получаемые при помощи этих алгоритмов, являются эстетически приятными, показывают симметрию и порождают (если это возможно) размещения без пересечений ребер
- 5) Их легко настраивать на новые приложения

Методы размещения, основанные на физических аналогиях

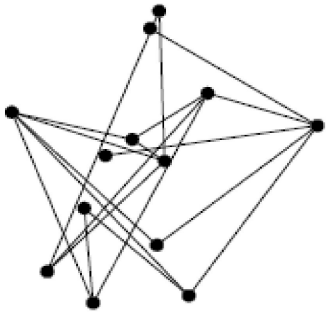
- Основу любого силового алгоритма составляют две компоненты:
- модель, описывающая физические объекты (соответствующие вершинам и ребрам графа) и взаимодействие между этими объектами
- алгоритм, который (приблизительно) вычисляет состояние равновесия для этой системы
- Описание модели основывается на том, какое изображение можно считать хорошим в каждом конкретном случае.
- С моделью связывается целевая функция, описывающая конкретное понятие «хорошести»
- Алгоритм служит для оптимизации целевой функции.

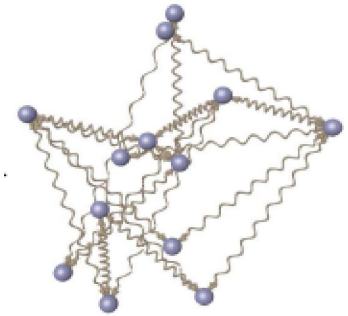
Методы размещения, основанные на физических аналогиях (общие рассуждения)

- Рассмотрим, к примеру, связный неориентированный граф $G(V,E)$ и попробуем получить прямолинейное изображение этого графа, обладающее следующими свойствами:
- 1) Вершины равномерно распределены на поверхности изображения.
- 2) Смежные вершины (соединенные ребром) должны быть расположены примерно на одинаковом расстоянии друг от друга.

Методы размещения, основанные на физических аналогиях (общие рассуждения)

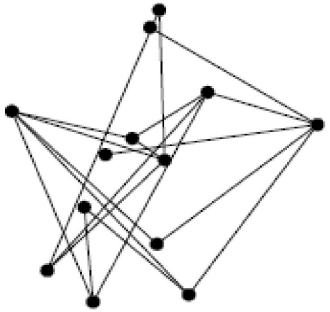
- Эти пожелания можно обосновать только интуитивно.
- Равномерное распределение вершин уменьшает беспорядок,
- а одинаковая длина ребер дает впечатление неискаженного изображения.
- Поскольку понятия «искажение» и «беспорядок» уже имеются в физике, можно пообсуждать физические аналогии, где встречаются такие понятия.
- Равномерное распределение можно искать для движущихся объектов.
- Достаточно естественно представить вершины как заряженные шары, которые отталкиваются друг от друга для удовлетворения первого критерия.
- А чтобы смежные вершины не разбегались слишком далеко, их можно соединить чем-то, например, пружинами, соответствующими ребрам.

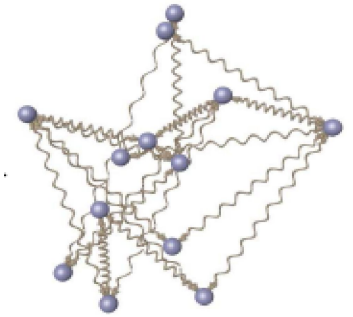




Методы размещения, основанные на физических аналогиях (общие рассуждения)

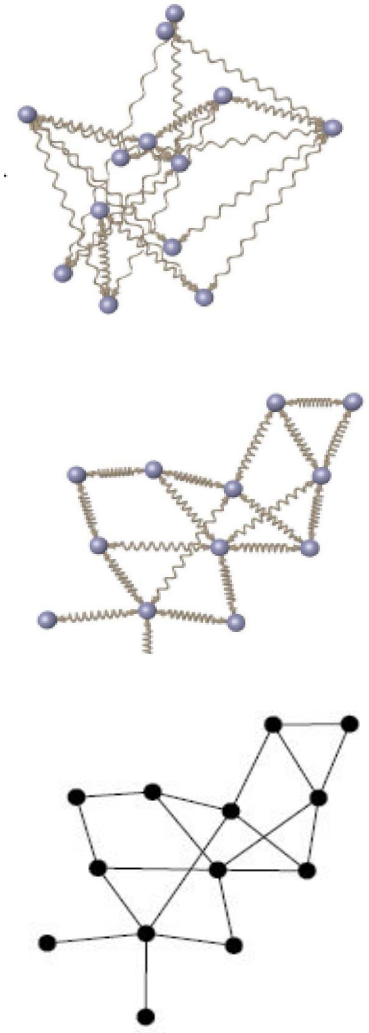
- Пружины подходят больше, чем жесткие палки или веревки, так как они могут и удлиняться и сжиматься. Чем сильнее отклонение от «естественной длины» тем больше будет сила, действующая на них, чтобы вернуть пружины к заданной длине. При этом небольшое искажение естественной длины неизбежно, так как чаще всего невозможно изобразить весь граф с прямыми ребрами одинаковой длины.
- Доказано, что проблема выяснения имеет ли произвольный граф прямолинейное размещение с ребрами равной длины в любом количестве измерений является *NP*-полной (1982, Джонсон).





Методы размещения, основанные на физических аналогиях (общие рассуждения)

- Если система описана, и объектам разрешено двигаться под влиянием действующих на них сил, то она постепенно придет в состояние равновесия , в котором все силы уравновешивают друг друга и взяв изображение, соответствующее этому положению равновесия, мы получим изображение, приблизительно удовлетворяющее указанным выше критериям.



Силовые алгоритмы (Spring embedder, Eades 1984)

Дан связный неориентированный граф G = (V, E)

- Пусть P = (p_v), v∈V – это вектор позиций вершин. Каждая вершина v имеет координату $p_v = (x_v, y_v)$.
- Расстояние между вершинами $||p_v - p_u||$ - это Евклидово расстояние.
- Будем обозначать

$$\overrightarrow{p_u p_v} = \frac{p_v - p_u}{|| p_v - p_u ||}$$

- Единичный вектор, направленный от p_u к p_v.
- Эстетичность размещения описывалось словами: «все ребра должны быть одинаковой длины, а размещение должно быть как можно более симметричным.

Силовые алгоритмы (Eades 1984)

- 1) Сила отталкивания действует между каждой парой не-смежных вершин: u, v ∈V

$$f_{rep}(p_u, p_v) = \frac{c_{rep}}{\| p_v - p_u \|^2} \overrightarrow{p_u p_v}$$

Где c_{rep} является константой

2) Дополнительно, силы пружины между смежными вершинами будут держать их близко друг к другу.

$$f_{spring}(p_u, p_v) = c_{spring} \log \frac{\| p_u - p_v \|}{l} \overrightarrow{p_v p_u}$$

Где c_{spring} - это параметр, управляющий силой пружины

l – « естественная» длина пружины

Силовые алгоритмы (Eades 1984)

Алгоритм Spring embedder (Eades 1984)

Вход: связный неориентированный граф G = (V, E) и начальное размещение его вершин p = (p_v) v∈V

Выход: Размещение с низким внутренним напряжением

for t :=1 **to** *Количество_итераций* **do**

for v∈ V **do{**

$$F_v(t) < - \sum_{u:\{u,v\}\notin E} f_{rep}(p_u, p_v) + \sum_{u:\{u,v\}\in E} f_{spring}(p_u, p_v)$$

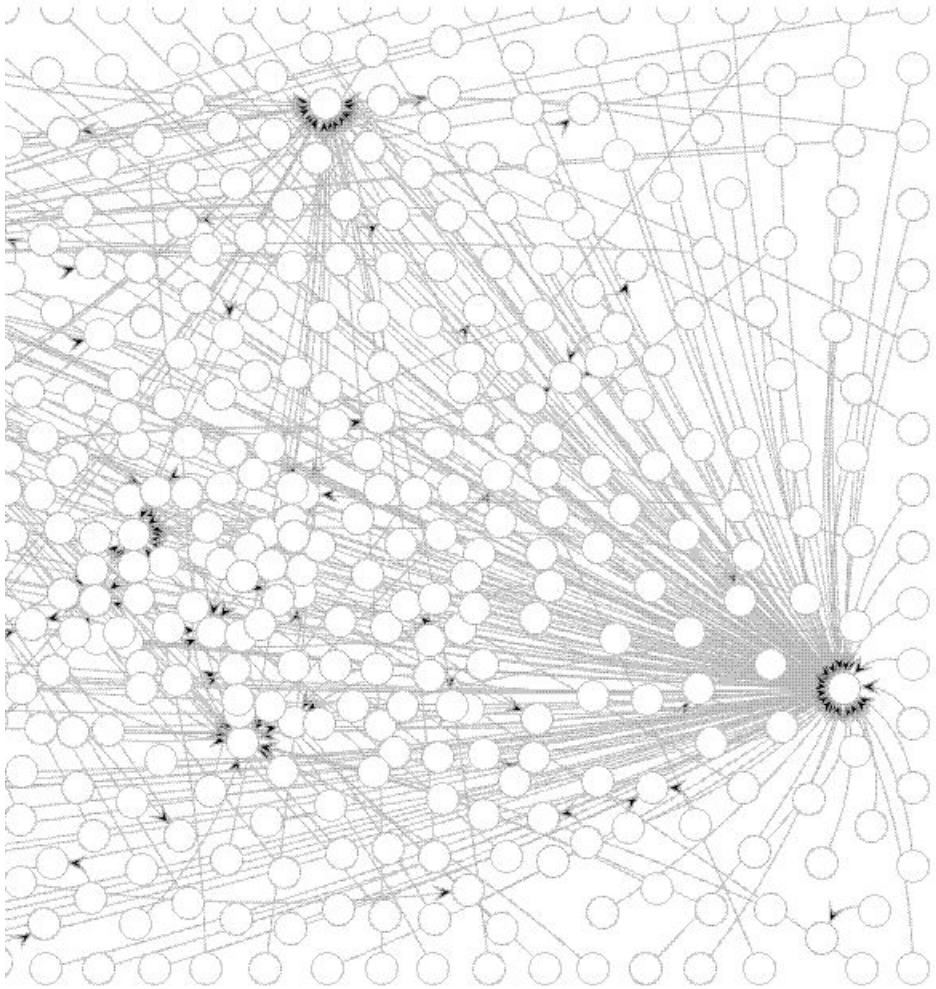
}

for v∈ V **do** $p_v := p_v + \delta F_v(t)$

}

У Идеса: C_{spring} = 2, C_{rep} =1, l =1, δ =0.1, кол-во итераций =100.

Пример изображения, порождаемого алгоритмом Eades



Силовые алгоритмы (Fruchterman &Reingold, 1991)

- Алгоритм Фрюхтермана и Рейнгольда 1991 года ввел критерий равномерного распределения и попробовал его формализовать.
- Также ввел несколько модификаций, направленных, в основном, на ускорение работы алгоритма, так как результаты, получаемые этим алгоритмом, весьма похожи на результаты работы алгоритма Идеса.

Fruchterman&Reingold(1991). Модификация сил, действующих на вершины

Силы отталкивания действуют между **каждой** парой вершин

$$f_{rep}(p_u, p_v) = \frac{l^2}{\| p_u - p_v \|} \overrightarrow{p_u p_v}$$ **для всех (u, v)∈V**

Силы притяжения действуют только между смежными вершинами.

$$f_{attr}(p_u, p_v) = \frac{\| p_u - p_v \|^2}{l} \overrightarrow{p_v p_u}$$

Сила пружины вычисляется как сумма этих сил

$F_{spring}(p_u,p_v) = f_{attr}(p_u,p_v)+f_{rep}(p_u,p_v)$

Fruchterman&Reingold(1991).

- Сила притяжения вычисляется быстрее, потому что не надо вычислять корень.
- Процесс быстрее сходится к решению, за счет того, что увеличено влияние компоненты расстояния (квадратный показатель степени).

Силовые алгоритмы (Fruchterman &Reingold, 1991)

1. **area:= W * L; {W и L – это ширина и длина фрейма}**
2. **G := (V, E); вершинам присваиваются случайные начальные позиции**
3. **I := √area/|V|;/* идеальная длина зависит от площади и количества вершин*/**
4. **function f_{rep} (x) := begin return I^2/x end;**
5. **function f_{atr}(x) := begin return x^2/I end;**
6. **for i := 1 to iterations do begin**

Силовые алгоритмы (Fruchterman &Reingold, 1991)

/* вычислить силы отталкивания, действующие на каждую вершину со стороны всех остальных вершин и смещение вершины под влиянием силы отталкивания: */

- **for v in V do begin {**
  - **/* каждая вершина имеет два вектора: .pos и .disp */**
  - **v.disp := 0;**
  - **for u in V do {**
    - **if (u ≠v) then begin**
    - **/* δ-это вектор разностей между позициями двух вершин*/**
    - **δ:= v.pos – u.pos;**
    - **v.disp := v.disp +(δ/|δ|) * f_{rep} (|δ|)**

      }
- **}**

Силовые алгоритмы (Fruchterman &Reingold, 1991)

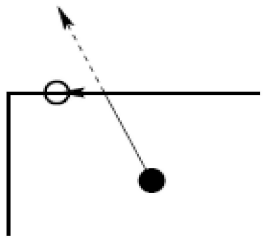
- **/*вычислить силы притяжения между двумя смежными вершинами смещения обеих вершин под влиянием силы притяжения:*/**
- **for e in E do {**
- **/*каждое ребро это упорядоченная пара вершин .v и .u*/**
- **δ := e.v.pos – e.u.pos;**
- **e.v.disp := e.v.disp - (δ/|δ|) . f_{atr}(|δ|);**
- **e.u.disp := e.u.disp +(δ/|δ|) . f_{atr}(|δ|);**
- **}**

Силовые алгоритмы (Fruchterman &Reingold, 1991)

- **/*ограничить max_смещение температурой t и предотвратить перемещения за границу фрейма*/**
- **for v in V do {**
- **v.pos := v.pos +(v.disp/[v.disp|) * min(v.disp, t);**
- **v.pos.x := min(W/2, max(-W/2, v.pos.x));**
- **v.pos.y := min(L/2, max(-L/2, v.pos.y))**
- **}**
- **/*уменьшить температуру по мере приближения размещения к лучшей конфигурации*/**
- **t := cool(t)**
- **}**

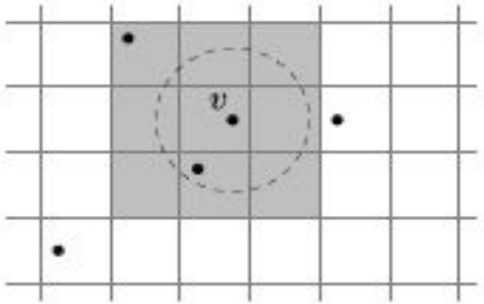
Модификации, связанные с вектором смещения

1) вместо использования константного множителя δ ввели смещение, зависящее от температуры δ(t) и постепенно убывающее, чтобы избежать излишних перескоков вершин, особенно на заключительных этапах
2) Для того, чтобы вершины не выскакивали из заданной области, производится обрезка вектора, оставляющая вершину на границе области



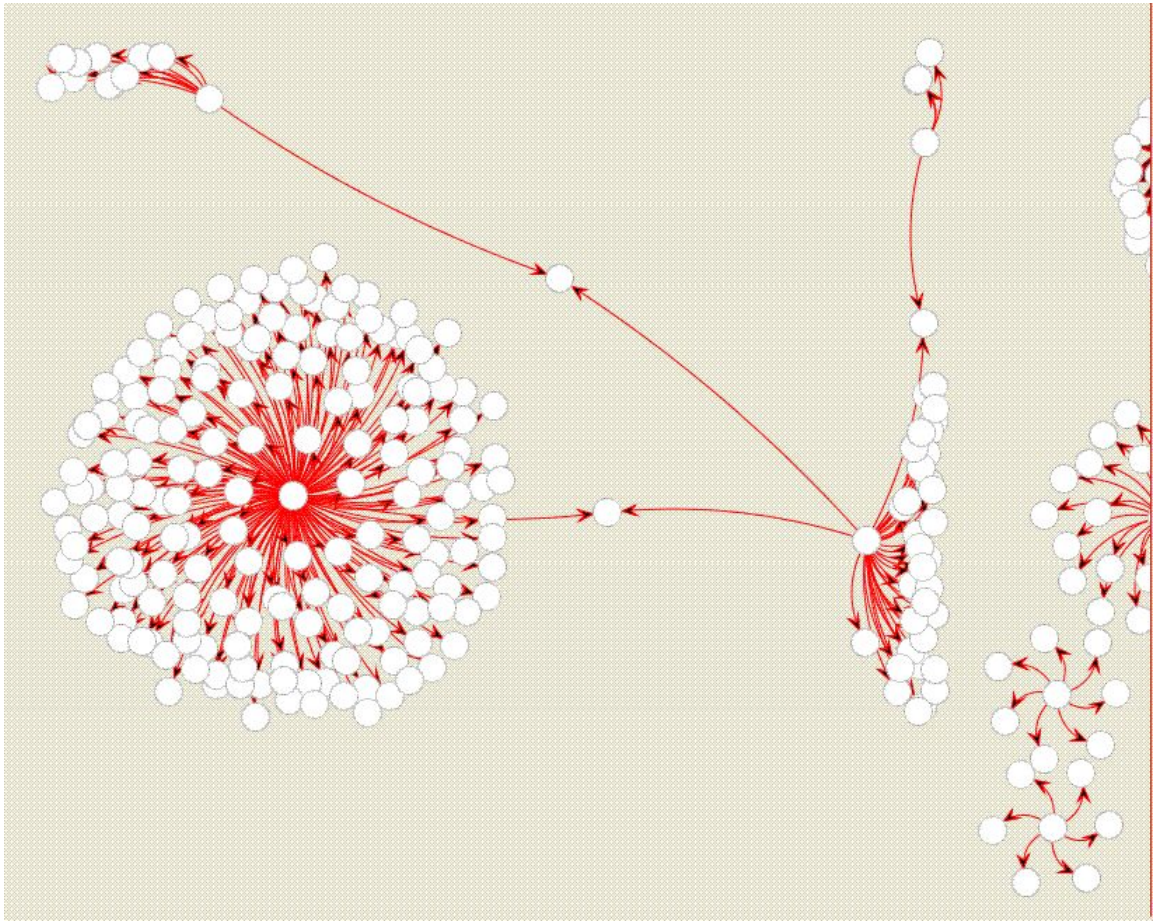
Fruchterman&Reingold(1991)

На каждой итерации базовый алгоритм вычисляет O(|E|) сил притяжения и O($|V|^2$) сил отталкивания. Для уменьшения квадратичной сложности сил отталкивания Фр&Ре предложили использовать сеточный вариант их базового алгоритма, где силы отталкивания между отдаленными вершинами игнорируются.



- Поскольку отталкивание отдаленных вершин не сильно влияет на вектор смещения, то эти несущественные вершины отбрасываются из суммы сил отталкивания. Рассматриваются только вершины, расположенные в узлах сетки, близких к узлу *v* и, если их расстояние меньше заданного порога, только тогда вычисляется сила отталкивания. Это позволяет оценку по времени *O(n)* для вычисления сил отталкивания.

Пример изображения, получаемого алгоритмом Fruchterman- Reingolda



Силы гравитации и алгоритм Frick

- Одной пружинной модели оказалось недостаточно, поскольку обнаружилось, что в случае несвязного графа или слабо связанных компонент эти несвязанные компоненты будут разлетаться в разные стороны из-за недостатка сил притяжения. Эти слабосвязанные компоненты размещаются далеко друг от друга так что ребра между ними неэстетично длинны.
- Спорный вопрос!

Силовые алгоритмы (Frick et al,1995)

- Поэтому в работе (Frick et al 1995) была введена еще дополнительная сила: сила гравитации, которая зависит от количества ребер, инцидентных вершине v, то есть, от степени вершины v.
- Другое заметное улучшение касается ускорения сходимости алгоритма. Силы отталкивания и притяжения были изменены так, чтобы не надо было вычислять квадратный корень:

Силовые алгоритмы (Frick et al,1995)

- То есть, вершины с высокой степенью двигаются медленнее, они ведь «тяжелые»

$$f_{rep}(p_u, p_v) = \frac{l^2}{\| p_v - p_u \|^2} \overrightarrow{p_u p_v}$$

$$f_{attr}(p_u, p_v) = \frac{\| p_u - p_v \|^2}{l \cdot \Phi(v)} \overrightarrow{p_v p_u}$$

$$\Phi(v) = 1 + \frac{\mathbf{deg}(v)}{2}$$

Силовые алгоритмы (Frick et al,1995)

- Кроме того, была введена сила гравитации, которая толкает каждую вершину к общему центру тяжести.

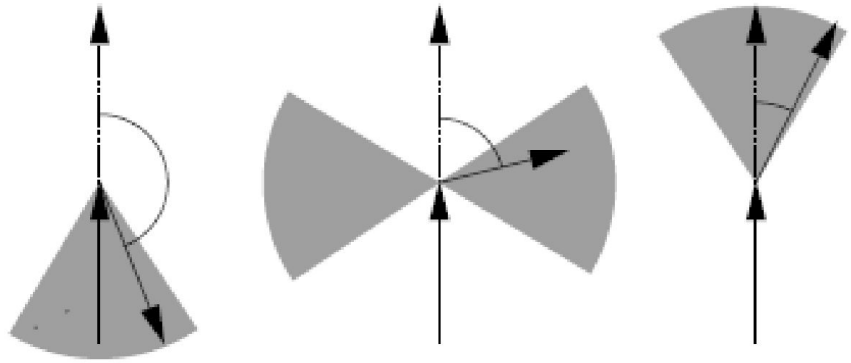
$$B = \frac{1}{|V|}\sum_{w \in V} p_w$$

$$F_{grav}(v) = \Phi(v) \cdot c_{grav} \cdot (\overrightarrow{B - p_v})$$

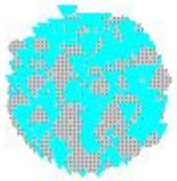
$$\Phi(v) = 1 + \frac{\mathbf{deg}(v)}{2}$$

Силовые алгоритмы (Frick et al,1995)

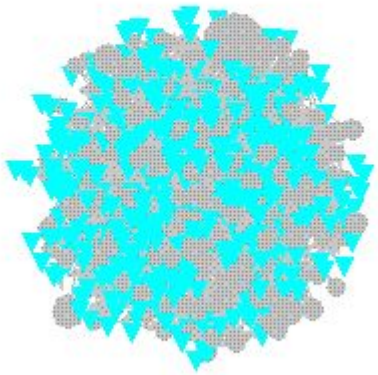
- Чтобы уменьшить количество итераций, вектор $F_v(t-1)$ запоминался и сравнивался с $F_v(t)$
- То есть вычислялся угол $\alpha = \angle (F_v(t-1), F(t)$. Если угол небольшой, то есть движение происходит в том же самом направлении, то $\delta_v(t)$ выбирался побольше, а если угол большой, то есть направление движения вершины менялось, то $\delta_v(t)$ выбирался поменьше.
- Также подсчитывалась мера поворота, если угол $\alpha = \angle(F_v(t-1), F_v(t)$ близок к 90°, то $\delta_v(t)$ тоже понижалось.



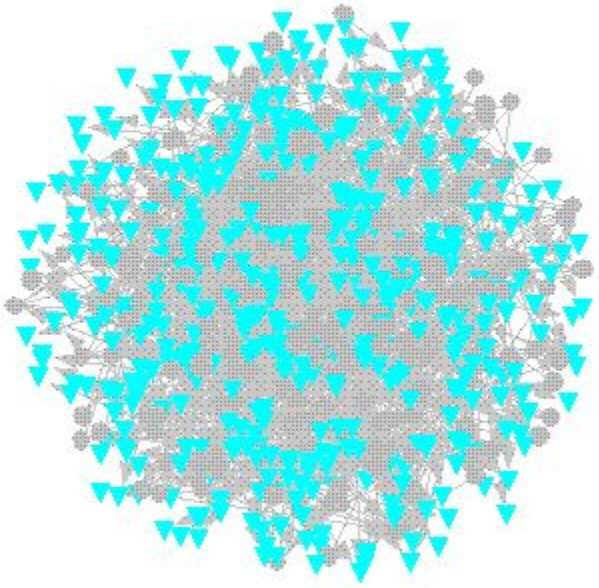
Силовые алгоритмы (Frick et al,1995)



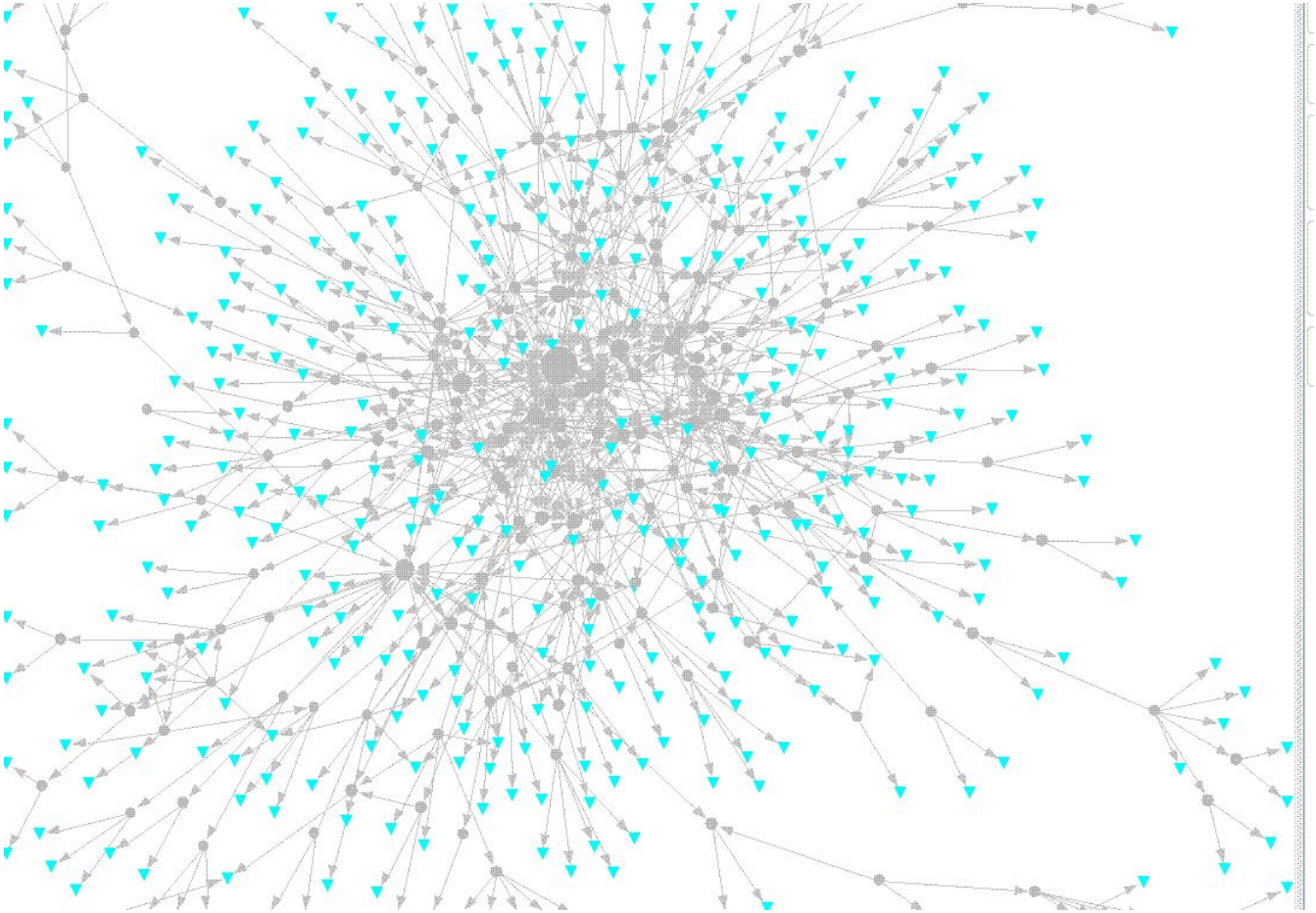
Силовые алгоритмы (Frick et al,1995)



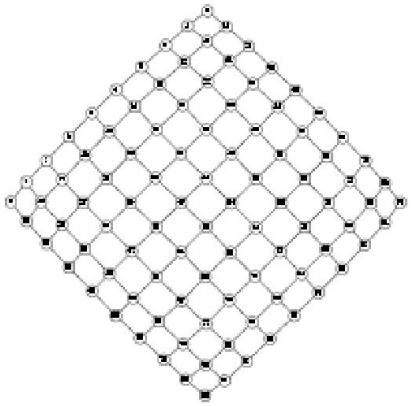
Силовые алгоритмы (Frick et al,1995)



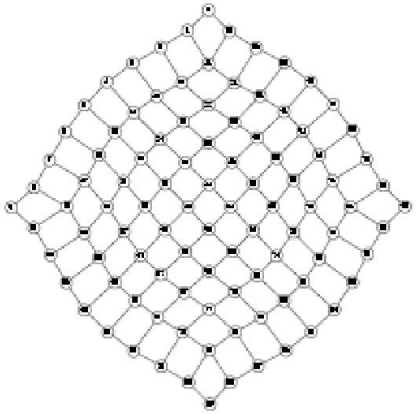
Силовые алгоритмы (Frick et al,1995)



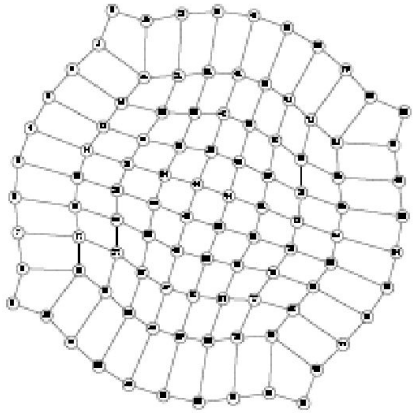
Силовые алгоритмы, сила гравитации



Нет силы гравитации



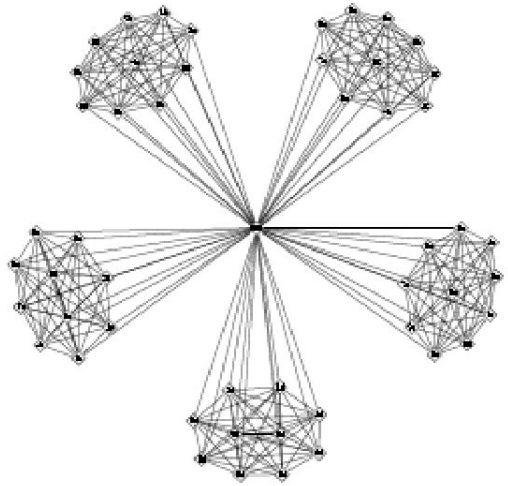
коэфф. гравитации = 0,6



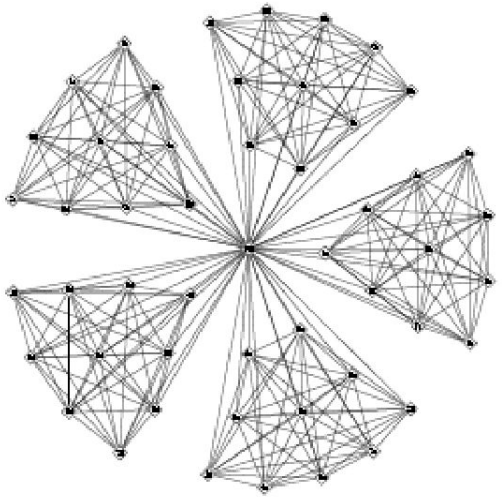
Коэфф. гравитации = 1,5

Поскольку силы гравитации направлены к центру тяжести, они налагают круговую структуру размещения

Силовые алгоритмы, сила гравитации



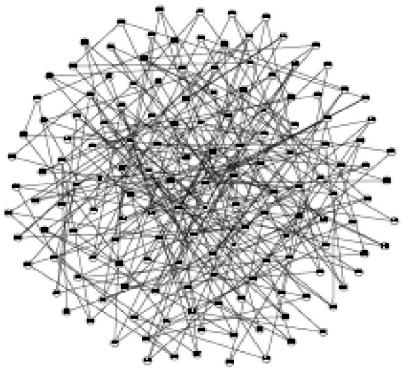
нет силы гравитации



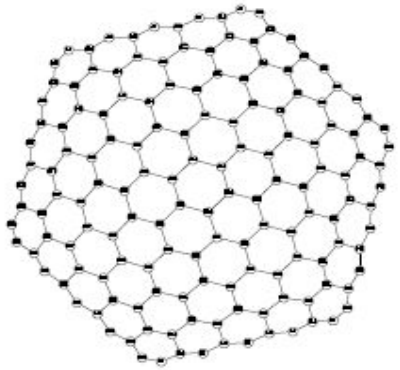
коэфф. гравитации = 2,0

Специфика гравитации становится заметна, если в графе есть слабо связанные плотные части. В этом случае без гравитации длина ребер сильнее различается, чем при гравитации.

Силовые алгоритмы



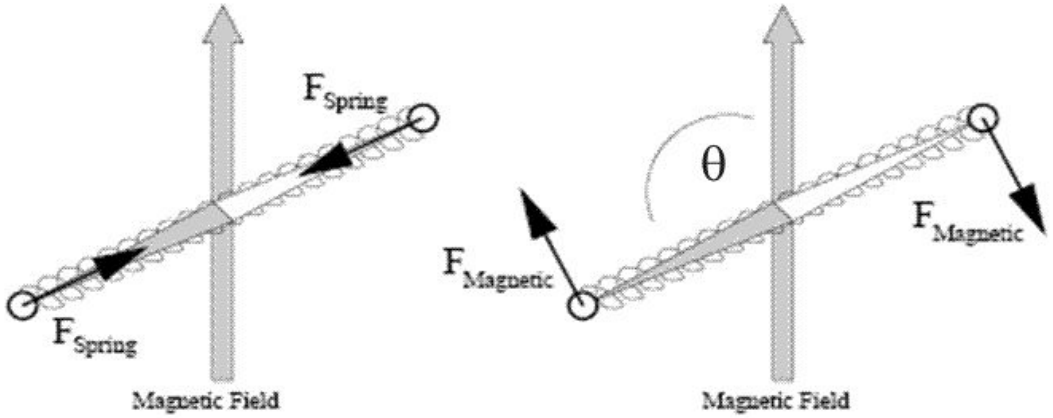
Если есть сила гравитации и отталкивания, вершины равномерно распределяются вокруг центра тяжести, но при этом совершенно не просматривается регулярность, индуцируемая структурой ребер. Результат работы силового алгоритма с использованием силы гравитации и отталкивания зарядов без использования пружинного притяжения



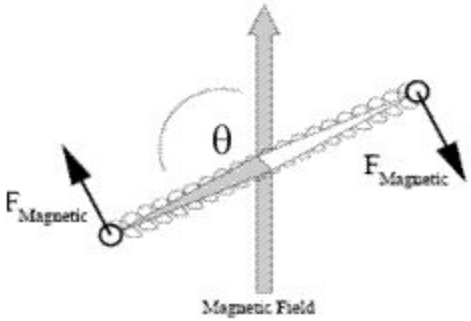
Результат работы силового алгоритма с использованием силы гравитации, силы отталкивания и силы притяжения пружин

Магнитное поле.

Sugiyama, Misue “A simple and unified method for drawing graphs: magnetic-spring algorithm”1995.
Пружинные алгоритмы не берут в расчет направления ребер.
В ориентированных графах желательно, чтобы все ребра были направлены в одну сторону.
Для решения этой проблемы Sugiyama, Misue предложили модель пружины, которая одновременно является магнитом и может вращаться в магнитном поле, как стрелка компаса.



Магнитное поле.



Сила магнитного поля, действующая на ребро, соединяющее вершины u и v , определяется по формуле:

$$F_m(u, v) = c_m\, d(u, v)^{\alpha}\, \theta^{\beta} \xrightarrow[p_u p_v]{} \perp$$

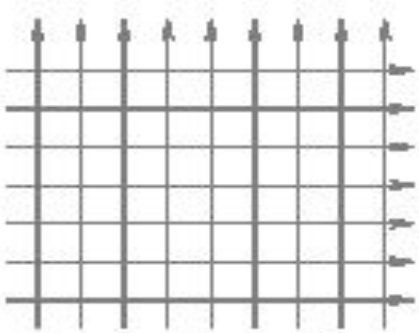
c_m – константа, управляющая силой магнитного поля
θ - угол между текущим направлением ребра (u, v) и направлением магнитного поля

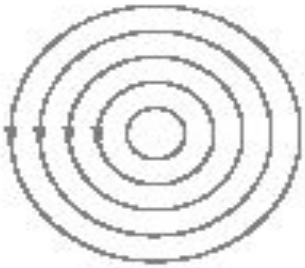
$\xrightarrow[p_u p_v]{} \perp$ - это вектор единичной длины, перпендикулярный вектору $\xrightarrow[p_u p_v]{}$ и направленный в сторону уменьшения угла θ.

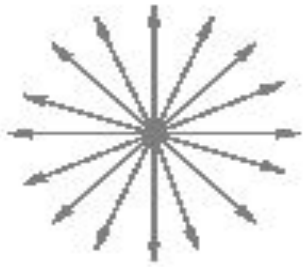
- $c_m, \alpha, \beta > 0$ параметры настройки системы

Магнитные поля [SM95]

- Разные типы магнитных полей:
- Параллельное: все силы действуют в одном направлении, может быть полезно для получения изображений, направленных сверху вниз
- Концентрическое: сила действует по концентрическим окружностям, выделяет циклы
- Радиальное: сила действует радиально из некоторой точки



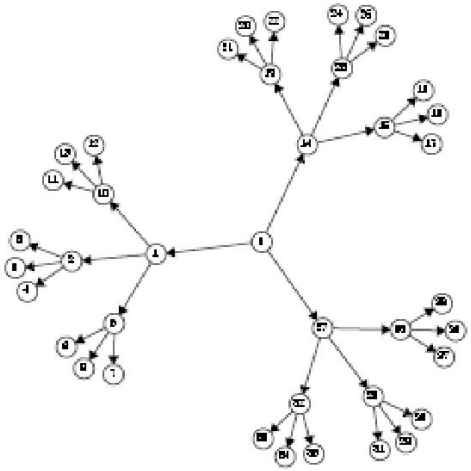




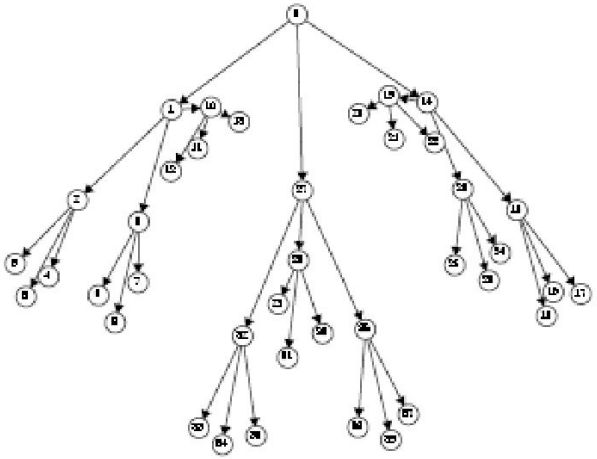
- Магнитные силы комбинируются с электрической силой и силой пружины
- Алгоритм поиска равновесия
  - Случайное начальное размещение и на каждой итерации смещать вершину в позицию с более низкой энергией

  Эта стратегия:
  - Может размещать ориентированные графы (однонаправленные пружины с одним из трех полей)
  - Может размещать ортогональные изображения, если применять комбинацию горизонтального и вертикального поля, а вершинам позволить принимать два направления
  - Может размещать изображения с линиями под углом 45° (карты железных дорог, метро, путей)
  - Успешно применяется к смешанным графам (графы, которые имеют одновременно и ориентированные и неориентированные ребра)

Влияние магнитного поля

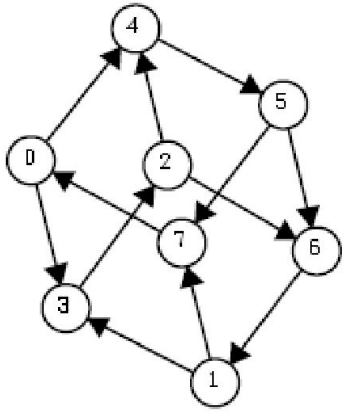


Нет магнитного поля

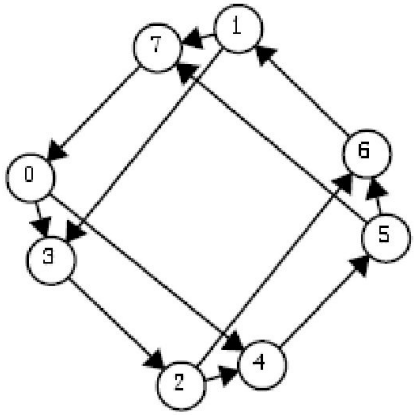


Параллельное магнитное поле

Влияние магнитного поля

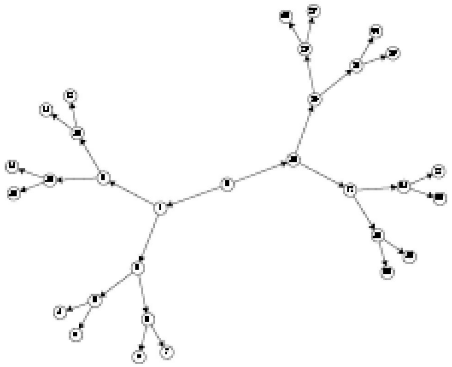


Нет магнитного поля

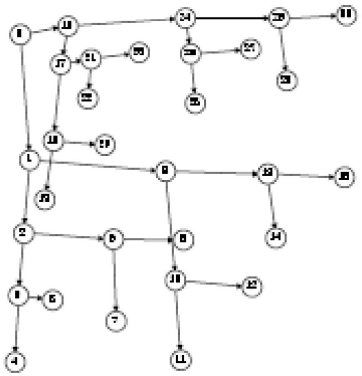


концентрическое магнитное поле

Влияние магнитного поля

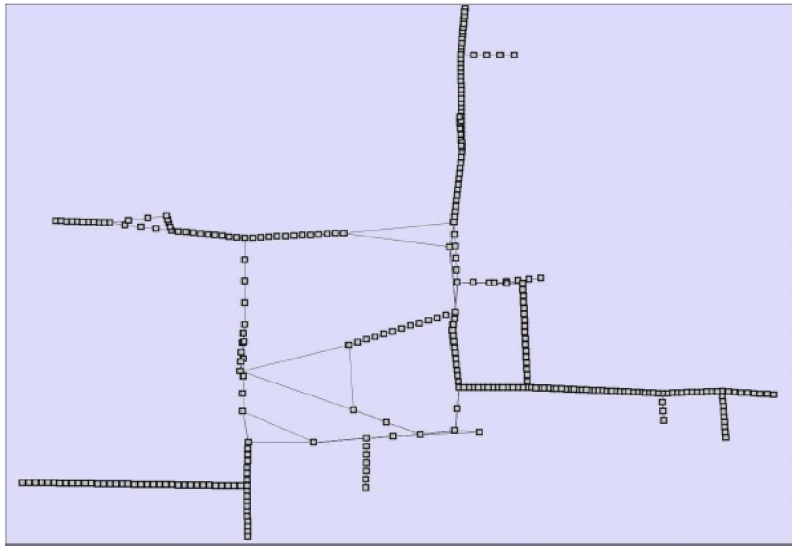


Нет магнитного поля

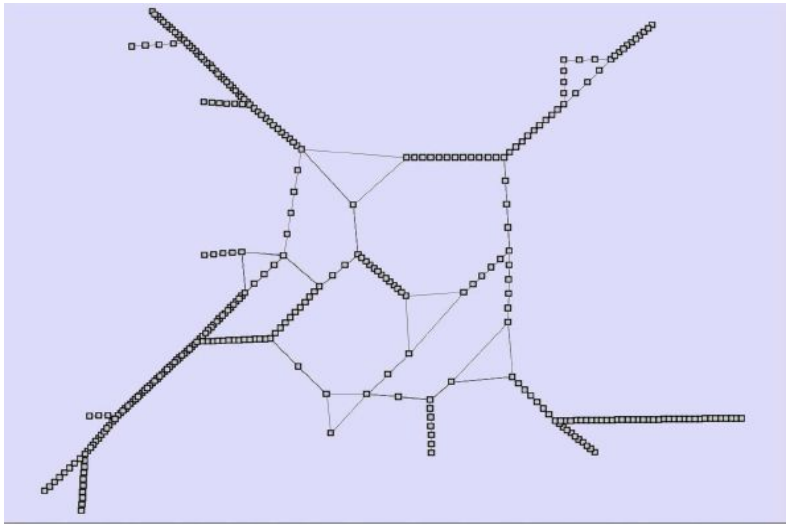


ортогональное магнитное поле

S.Hong, D.Merrick H.Nascimento Automatic Visualization of Metro Maps, 2006



План метро Сиднея, с применением ортогонального магнитного поля



План метро Сиднея с применением магнитного поля под углом 45 градусов

Размещения, основанные на энергии

- Силы, определенные в предыдущих алгоритмах, указывают, в каком направлении надо сдвинуть вершину, чтобы уменьшить силы, действующие на нее.
- Вместо того, чтобы описывать силы, действующие на вершину, можно попробовать описать энергию и попробовать минимизировать эту энергию.

[Kamada, Kawai 89]

- В 1989 году Камада и Кавай ввели другой взгляд на то, что считать хорошим размещением.
- В то время как алгоритмы Идеса и Фрюхтермана-Рейнгольда стараются держать смежные вершины (связанные ребром), на одинаковом идеальном расстоянии, Камада и Кавай предложили рассматривать в качестве идеального расстояния между любыми вершинами соответствующее расстояние между ними по графу, вычисляемое как кратчайший путь между всеми парами вершин.
- Поскольку эта цель не всегда может быть достигнута для произвольных двумерных и трехмерных графов евклидова пространства, подход пытается привести систему пружин в такое состояние, что минимизация энергии системы соответствует минимизации разницы между Евклидовым расстоянием и расстоянием по графу.

[Kamada, Kawai 89]

- Была взята за основу потенциальная энергия пружины, имеющей естественную длину *l*, растянутой до длины *d*:
- $E_{KK} = k_{spring} \cdot (d - l)^2$

- В этой модели нет раздельных сил притяжения и отталкивания. Вместо этого, вершины притягиваются или отталкиваются в зависимости от того, больше или меньше евклидово расстояние между ними, чем расстояние по графу. Разница расстояний подсчитывается по полному графу.

[Kamada, Kawai 89]

- Пусть d_{ij} означает длину кратчайшего пути между вершинами *i* и *j* в графе *G*(*V*, *E*).
- *L* - это длина единичного ребра на дисплее.
- Тогда $l_{ij} = L*d_{ij}$ - это идеальная длина пружины, соединяющей вершины *i* и *j*.
- Камада и Кавай предложили использовать в качестве $L = L_0/\mathbf{max}_{i<j} d_{ij}$, *где* L_0 *- это длина стороны дисплея, а* $\mathbf{max}_{i<j} d_{ij}$ - это диаметр графа, то есть расстояние между двумя наиболее удаленными вершинами. Сила пружины между вершинами *i* и *j* определяется как

$$k_{ij} = \frac{K}{d_{ij}^2}$$

- где *K*- это константа.

[Kamada, Kawai 89]

- Таким образом, получаем следующую функцию энергии:

$$E_{KK} = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{1}{2} k_{ij} (\| p_i - p_j \| - l_{ij})^2 .$$

Координаты частицы p_i в Евклидовой плоскости определяются значениями x_i и y_i , что позволяет переписать функцию энергии следующим образом:

$$E_{KK} = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{1}{2} k_{i,j} ((x_i - x_j)^2 + (y_i - y_j)^2 + l_{i,j}^2 - 2 l_{i,j} \sqrt{(x_i - y_i)^2 + (y_i - y_j)^2}$$

[Kamada, Kawai 89]

- Требуется найти такие значения переменных, которые минимизируют функцию энергии $E_{KK}(x_1,x_2,...x_n,y_1,y_2,\quad,y_n)$.
- Поскольку известно, что в локальном минимуме все частные производные равны 0, это соответствует решению системы из $2n$ нелинейных уравнений.

[Kamada Kawai 89]

- Уравнение можно решить при помощи итеративного подхода
- На каждом шаге перемещается одна вершина, которая минимизирует энергию, в то время как остальные вершины остаются зафиксированными
- Выбирается вершина, на которую действует наибольшая сила, то есть на каждом шаге этого алгоритма выбирается частица p_m с наибольшим значением градиента Δm, где

$$\Delta_m = \sqrt{\left(\frac{\partial E_{KK}}{\partial x_m}\right)^2 + \left(\frac{\partial E_{KK}}{\partial y_m}\right)^2}$$

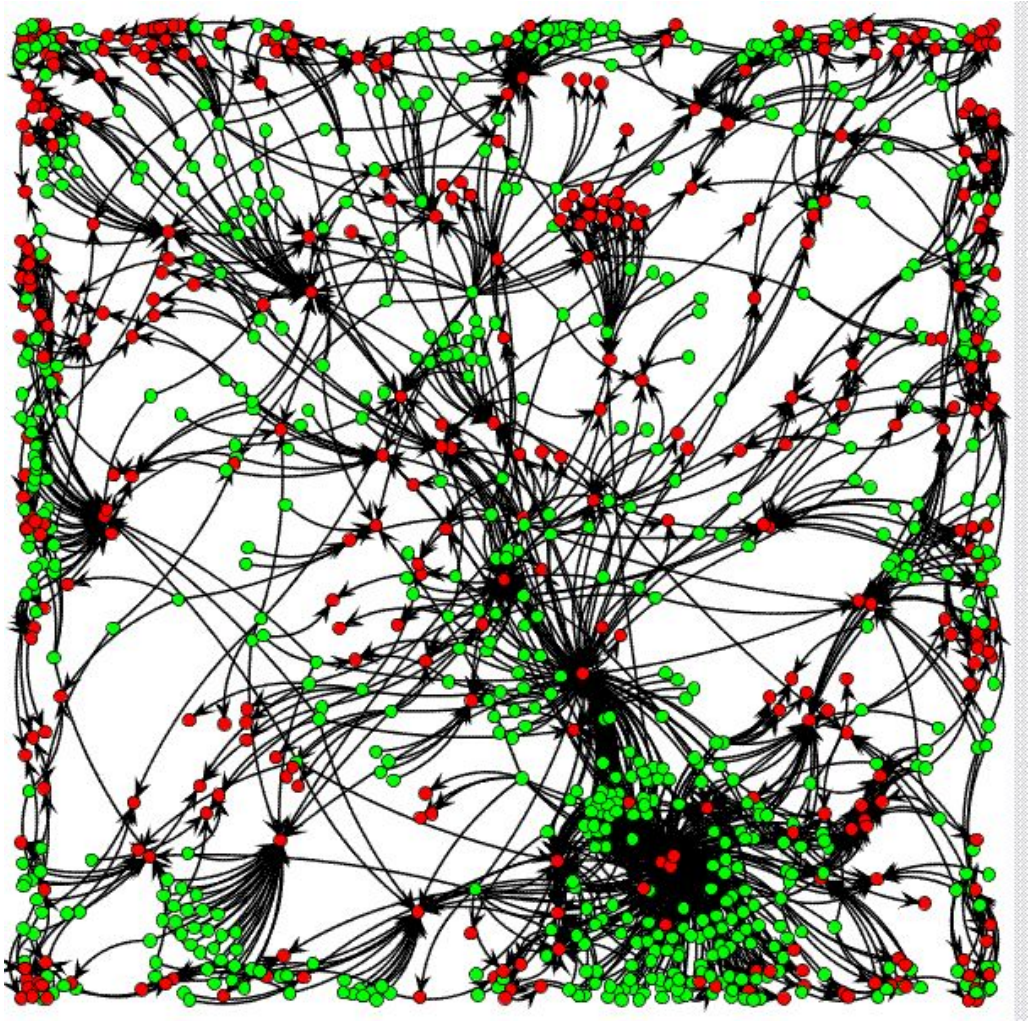
Все остальные вершины фиксируются, и энергия локально минимизируется перемещением одной вершины m

[Kamada, Kawai 89]

- Вычислить $d_{i,j}$ for $1 \le i \ne j \le n$;
- Вычислить $l_{i,j}$ for $1 \le i \ne j \le n$;
- Вычислить $k_{i,j}$ for $1 \le i \ne j \le n$;
- initialize $p_1, p_2, \ldots, p$;
- **while** ($\max_i \Delta_i > \varepsilon$){
  - Пусть p_m - это частица, удовлетворяющая $\Delta m = \max_i \Delta_i$;
  - while ($\Delta m > \varepsilon$){

    /*вычислить δx и δxy решением следующей системы уравнений: */
- $$\frac{\partial^2 E_{KK}}{dx_m^2}\left(x_m^{(t)}, y_m^{(t)}\right)\delta x + \frac{\partial^2 E_{KK}}{\partial x_m \partial y_m}\left(x_m^{(t)}, y_m^{(t)}\right)\delta y = -\frac{\partial E_{KK}}{\partial x_m}\left(x_m^{(t)}, y_m^{(t)}\right);$$
  $$\frac{\partial^2 E_{KK}}{dx_m \partial y_m}\left(x_m^{(t)}, y_m^{(t)}\right)\delta x + \frac{\partial^2 E_{KK}}{\partial y_m^2}\left(x_m^{(t)}, y_m^{(t)}\right)\delta y = -\frac{\partial E_{KK}}{\partial y_m}\left(x_m^{(t)}, y_m^{(t)}\right);$$
- $x_m := x_m + \delta_x$;
- $y_m := y_m + \delta_y$;
- }
- }

Пример размещения полученного методом Фрюхтерман-Рейнгольда



Ontology visualization

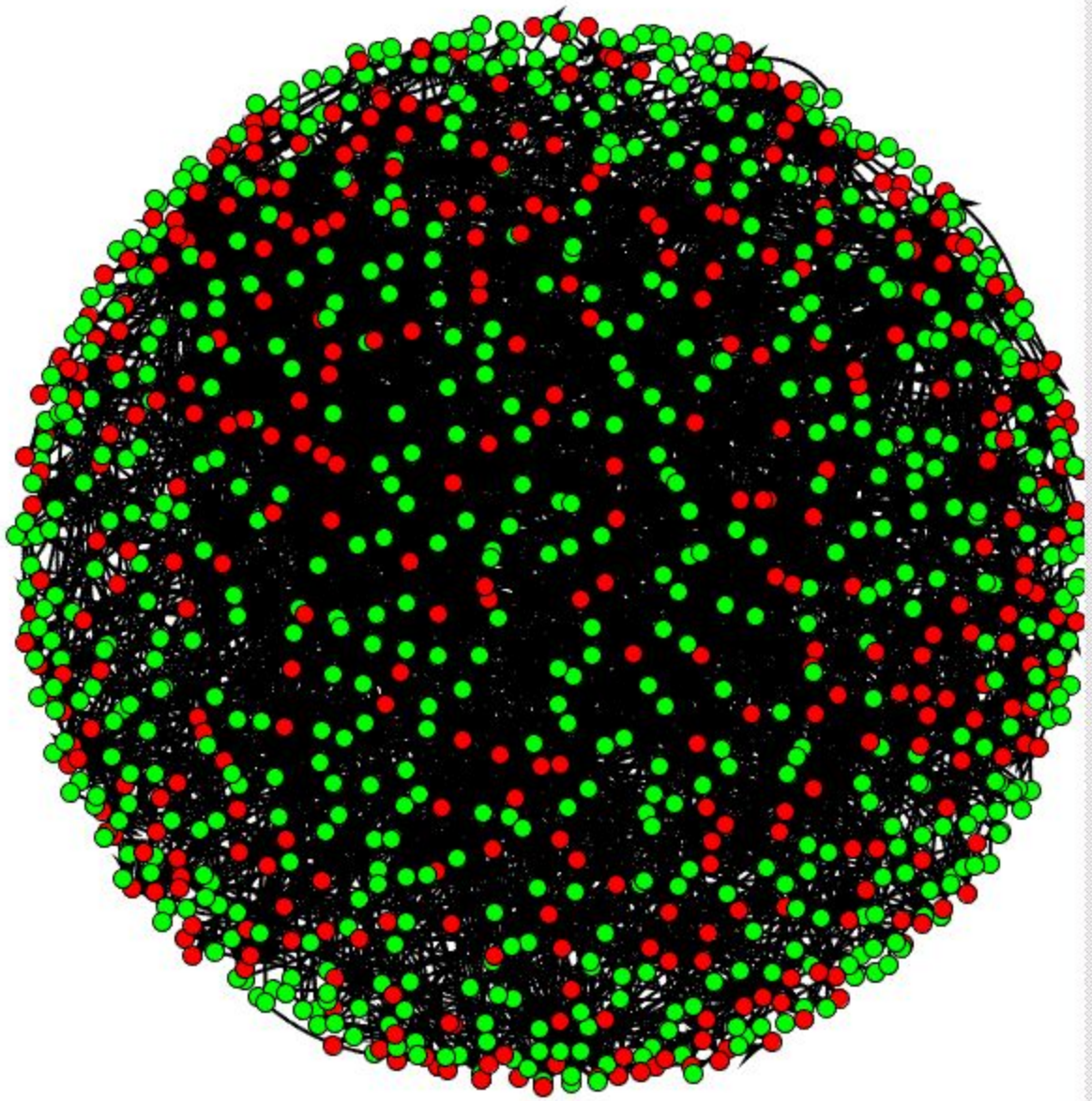
Classes | Relations | Associative and nested | Object relations | Class relations | Class sizes | Inheritance | Clustering

Visualize | Kamada-Kawai | TRANSFORMING | Object names

Choose relation

Публикация - Раздел науки(2107)
Автор(2090)
Направление-Исследований(1942)
Публикация_включает(1622)
Ресурс-Публикации(987)
Издан-в(901)
Раздел-Включает(749)
Изучает_Период(741)
Описывет класс методов(710)
Работает-в(601)
Ресурс-Человека(536)
Человек-проживает-в(460)
Применяет-метод(424)
Ресурс-Объекта(373)
Место-рождения(302)
Участник-События(290)
Описывает-Объект(277)
Объект-датирован(219)
Тематика Ресурса(157)
Метод-Включает(138)

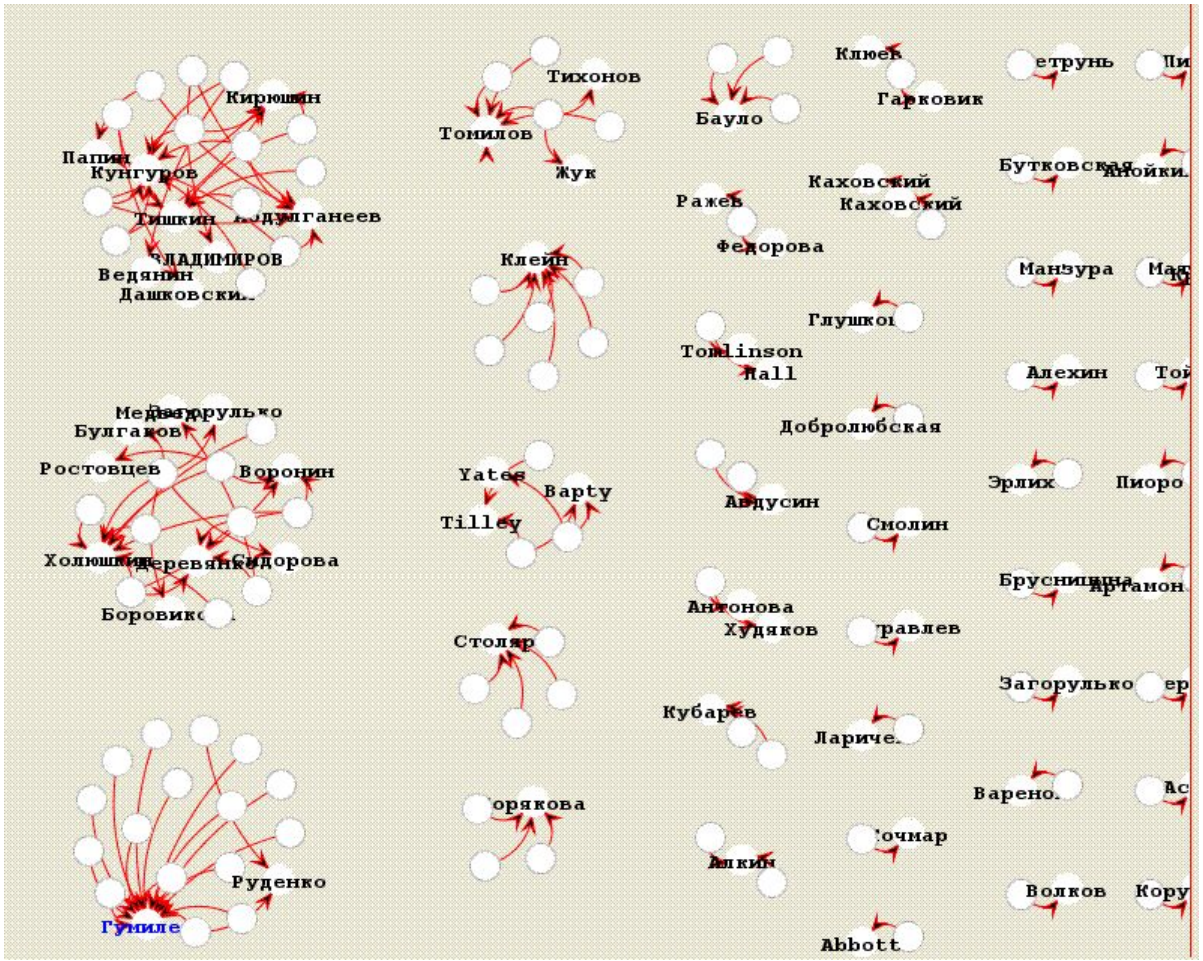
Description



Objects

1- Исследовате
1- Исследовате
1- Исследовате
1- Исследовате
1- Публикация -
1- Публикация -
1- Публикация -
1- Исследовате
1- Публикация -
1- Исследовате
1- Публикация -
1- Публикация -
1- Публикация -
1- Публикация -
1- Исследовате
1- Исследовате
1- Исследовате
1- Исследовате
1- Публикация -
1- Исследовате
1- Публикация -
1- Исследовате
1- Исследовате
1- Публикация -
1- Публикация -
1- Публикация -
1- Публикация -
1- Публикация -
1- Исследовате
1- Публикация -
1- Исследовате
1- Публикация -
1- Публикация -
1- Публикация -
1- Исследовате
1- Публикация -
1- Исследовате
1- Публикация -
1- Исследовате
1- Публикация -

Пример результата размещения методом Камада-Кавая графа связей Автор_публикации с предварительным выделением несвязных компонент



[Kamada, Kawai 89]

- Алгоритм Камада-Кавая является вычислительно затратным, поскольку требует вычисления кратчайших путей между всеми парами вершин, что может быть сделано за время $O(|V|^3)$.
- Кроме этого он требует $O(|V|^2)$ памяти для хранения попарных расстояний между вершинами.
- Несмотря на свою затратность, его достоинством является простое и понятное определение того, что является хорошим размещением.

Метод барицентров(алгоритм Tutte,63)

- Исторически, метод барицентровТатта 1963 года был первым силовым алгоритмом для получения изображения без пересечений ребер 3-связного планарного графа.
- В отличие от большинства силовых алгоритмов, Татт гарантировал, что результирующее изображение не будет иметь пересечений ребер и более того, все грани изображения будут выпуклыми.
- Идея состоит в том, что если некоторую грань планарного графа зафиксировать на плоскости, то можно найти подходящие позиции для оставшихся вершин решением системы линейных уравнений, где каждая вершина представляется в виде выпуклой комбинации позиций ее соседей.

Метод барицентров(алгоритм Tutte,63)

- Вместо пружин переменной длины, как у Камада – Кавая, Татт считает, что идеальная длина всех пружин равна 0.

$$U_{Tutte}(p) = \sum_{\{u,v\}\in E} \| p_u - p_v \|^2$$

Приравняв частные производные нулю, получаем две независимых системы линейных уравнений по одной на каждую координату.
Эти линейные системы могут быть переписаны в виде
$(D-A)\cdot x = 0$
$(D-A)\cdot y = 0$
Где A- это матрица смежности графа G , D- это диагональная матрица, где на диагонали стоят степени вершин, x и y –это вектора координат вершин. Матрица L= D-A называется Лапласиан.
Имеется тривиальное решение!

Метод барицентров(алгоритм Tutte,63)

- Разбить *V* на 2 подмножества: фиксированные вершины (не меньше 3) и свободные вершины
- Для достижения равновесия, выбрать такое размещение p_v, что $F_x(v) = 0$ для всех свободных вершин;
- Аналогично, выбрать такое размещение p_v, что $F_y(v) = 0$ для всех свободных вершин.

Поэтому,

$$F_x(v) = \sum_{(u,v)\in E}(x_v - x_u) = \deg(v)\cdot x_v - \sum_{w\in N_0(v)} x_w^* - \sum_{u\in N_1(v)} x_u = 0$$

- *deg(v)* – это степень вершины *v*,
  - $N_0(v)$: множество фиксированных вершин *v*;
  - $N_1(v)$: множество свободных соседей вершины *v*.

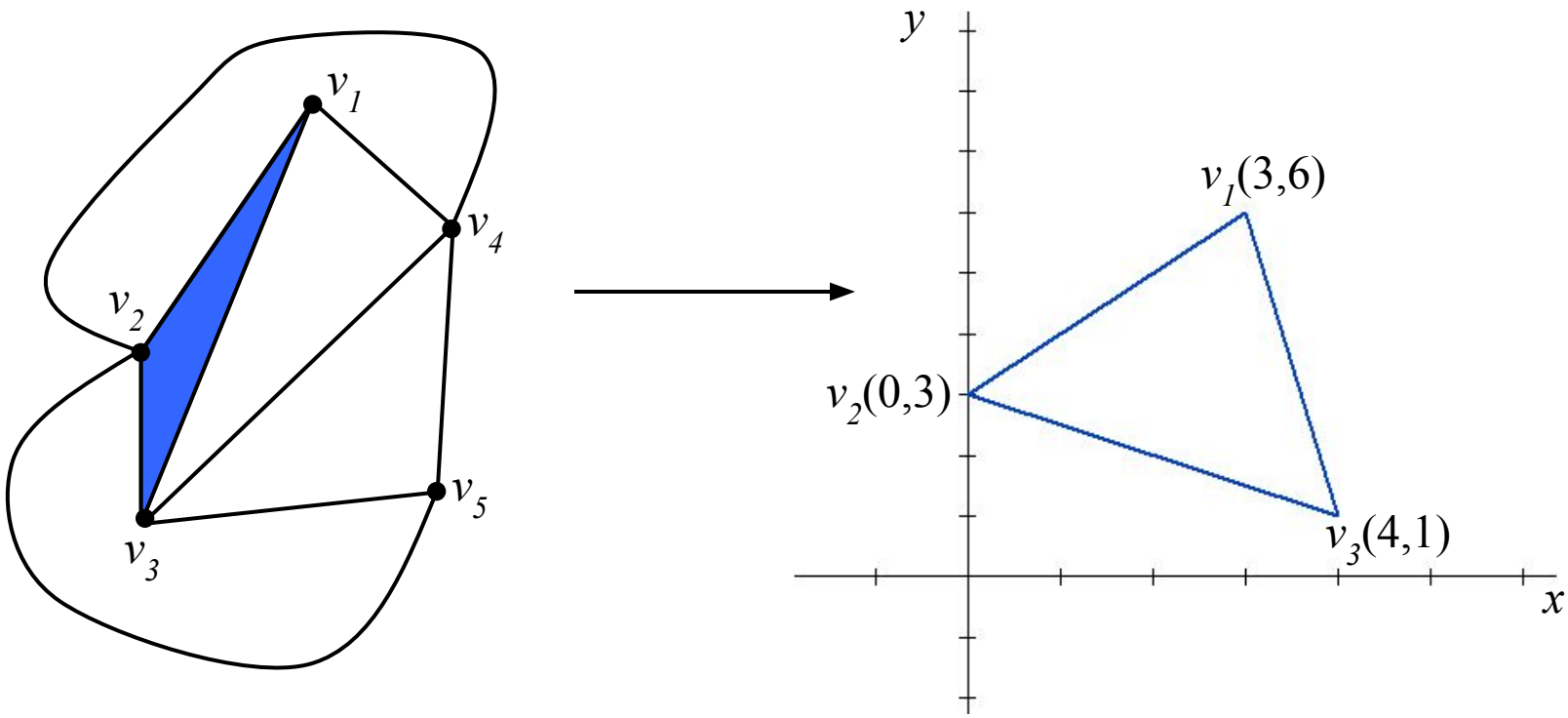
Метод барицентров(алгоритм Tutte,63)

$$\deg(v)x_v - \sum_{u \in N_1(v)} x_u = \sum_{w \in N_0(v)} x_w^*, \deg(v)y_v - \sum_{u \in N_1(v)} y_u = \sum_{w \in N_0(v)} y_w^*$$

- Все уравнения линейные.
- Количество уравнений и количество неизвестных равны количеству свободных вершин.
- Решается размещением каждой свободной вершины в барицентр ее соседей.
  - Отсюда и название «метод барицентров»

Пример

G:

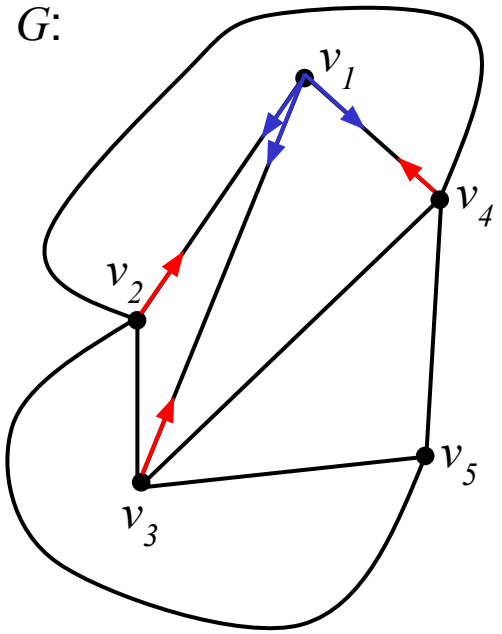


G - 3-связный граф

Сделаем грань $\{v_1, v_2, v_3\}$ внешней

Пример (2)

- $V(J) = \{v_1, v_2, v_3\}$
- $N(4) = \{v_1, v_2, v_3, v_5\}$
- $N(5) = \{v_2, v_3, v_4\}$
- $L(G) = \begin{bmatrix} 3 & -1 & -1 & -1 & 0 \\ -1 & 4 & -1 & -1 & -1 \\ -1 & -1 & 4 & -1 & -1 \\ -1 & -1 & -1 & 4 & -1 \\ 0 & -1 & -1 & -1 & 3 \end{bmatrix}$



Пример: Алгоритм Tutte

- Пусть координаты вершин имеют следующие значения:
- $v_1 = (3,6)$, $v_2 = (0, 3)$, $v_3 = (4, 1)$.
-
- Соответствующая система линейных уравнений для вычисления x-координат вершин v_4 и v_5 через известные x-координаты вершин v_1 v_2 и v_3 имеет вид:
- $L_{41} v_{1x} + L_{42} v_{2x} + L_{43} v_{3x} + L_{44} v_{4x} + L_{45} v_{5x} = 0$ (2)
- $L_{51} v_{1x} + L_{52} v_{2x} + L_{53} v_{3x} + L_{54} v_{4x} + L_{55} v_{5x} = 0$ (3)

Пример: Алгоритм Tutte

- В соответствии с нашим выбором $v_{1x} = 3$, $v_{2x} = 0$, $v_{3x} = 4$.
- L_{44} равно степени вершины *4,* то есть четырем, все остальные L_{4i} равны минус единице. L_{55} равно степени вершины *5,* то есть трем, L_{51} равно нулю, поскольку v_5 и v_1 не связаны ребром, все остальные L_{5i} равны минус единице. Подставив эти значения, получим два выражения:
- $-1 * 3 + (-1) * 0 + (-1) * 4 + 4\ v_{4x+}\ (-1) *\ v_{5x} = 0$
- $0 * 3 + (-1) * 0 + (-1) * 4 + (-1) *\ v_{4x\ +}\ 3 *\ v_{5x} = 0$
-
- В результате имеем систему уравнений:
- $4v_{4x} - 7 = v_{5x}$
- $-v_{4x} + 3\ v_{5x} = 4$
- Решением этой системы будут $v_{4x} = 25/11$, $v_{5x} = 23/11$.

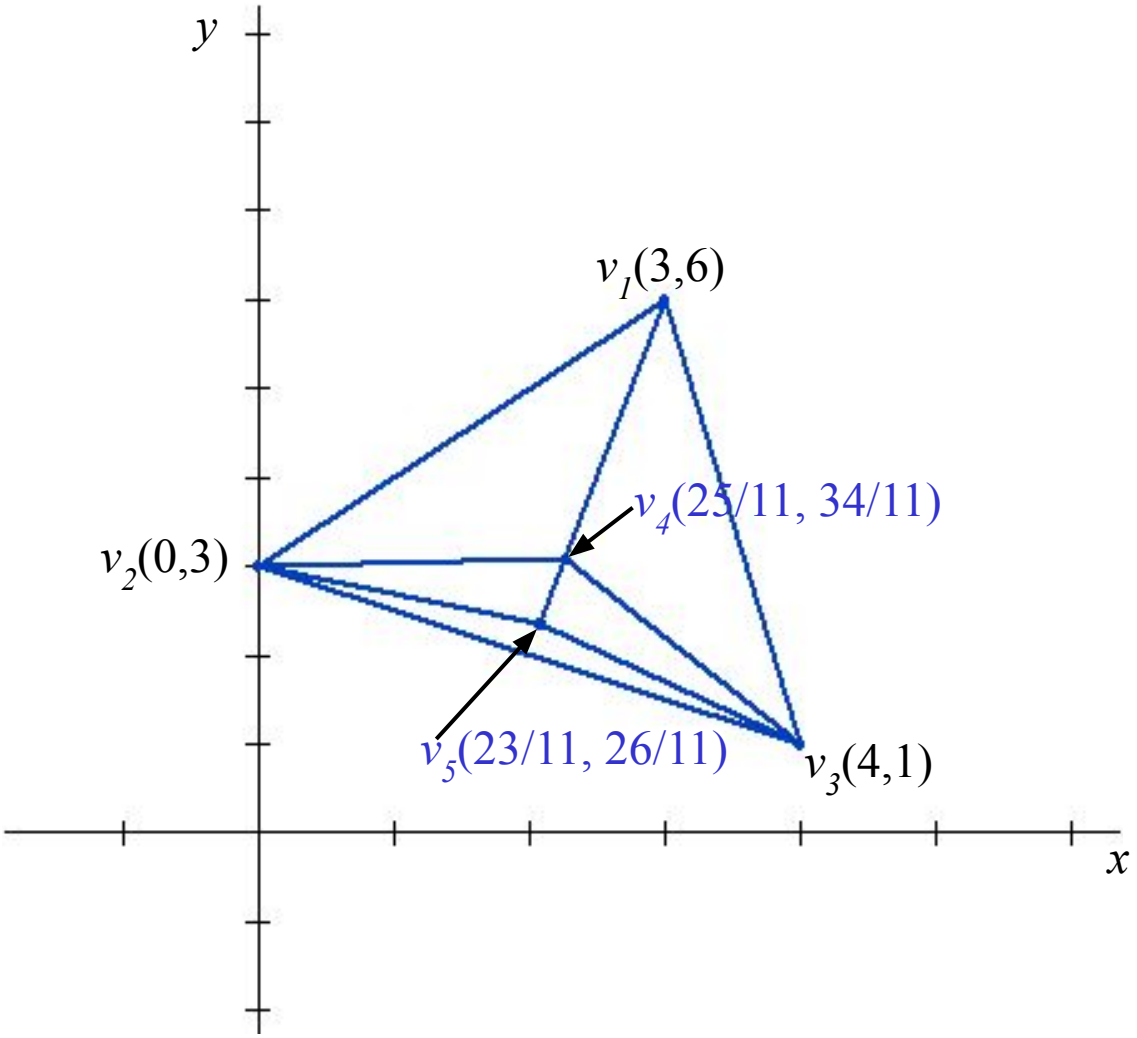
Пример: Алгоритм Tutte

Точно так же *y*-координаты вершин v_4 *и* v_5 вычисляются через известные *y*-координаты вершин v_1 , v_2 и v_3:

- $L_{41} v_{1y} + L_{42} v_{2y} + L_{43} v_{3y} + L_{44} v_{4y} + L_{45} v_{5y} = 0$
- $L_{51} v_{1y} + L_{52} v_{2y} + L_{53} v_{3y} + L_{54} v_{4y} + L_{55} v_{5y} = 0$
-
- Стало быть, система уравнений имеет вид:
- $4v_{4y} - v_{5y} = 10$
- $-v_{4y} + 3 v_{5y} = 4,$

- Решением системы будут $v_{4y} = 34/11$ и $v_{5y} = 26/11$.

Пример: Алгоритм Tutte



Algorithm Barycenter-Draw

Вход: разбиение множества вершин V,

V_0: не менее 3 фиксированных вершин

V_1: множество свободных вершин

Строго выпуклый многоугольник P с V_0 вершинами

Выход: размещение p_v

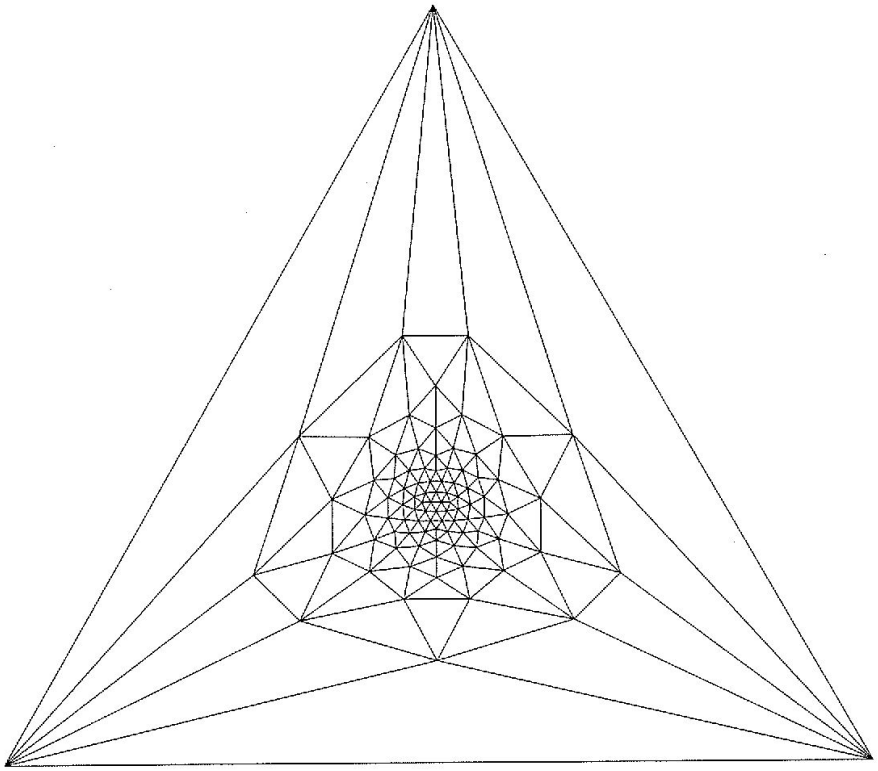
1. Поместить вершины $u \in V_0$ в вершины многоугольника P, а каждую свободную вершину в начало координат
2. repeat

- foreach свободной вершины $v \in V_1$ do
- $x_v = 1/\deg(v) \sum_{(u,v)\in E} x_u$
- $y_v = 1/\deg(v) \sum_{(u,v)\in E} y_u$
- until x_v and y_v сходятся для всех свободных вершин v.

Метод барицентров(алгоритм Tutte,63)

- Основная теорема, доказанная Tutte (1963) утверждает, что барицентрическое размещение 3-связных планарных графов является планарным, все грани являются выпуклыми, если вершины одной грани в единственной планарной укладке зафиксированы на границе выпуклого многоугольника в соответствующем порядке. Такое размещение может быть получено за время *O(nlogn).*

Пример размещения, полученного методом барицентров



1) Применим только для трехсвязных графов
2) Одним существенным недостатком этого метода является то, что результирующее изображение часто имеет плохую вершинную резолюцию.

Для любого n >1 существует граф такой, что метод барицентров вычисляет для него изображение экспоненциальной площади.

Ускорение силовых алгоритмов

- Экспериментальное сравнение базового алгоритма Fruchterman-Reingold, алгоритма GEM (Frick et al.,), метода the Kamada and Kawai, simulated annealing метода Davidson and Harel , осуществлялось Brandenburg et al. в 1995 году. Эксперименты показали, что все алгоритмы генерировали сравнительно хорошие размещения для небольших графов размера ($|V| + |E| < 180$ менее чем за 2 минуты.
- Для больших графов они не очень подходят. Например, GEM требовал в тот момент 71 секунду для построения изображения графа, содержащего 256 вершин и представлявшего собой регулярную квадратную сетку. Оценка времени работы для графа, содержащего 25600 вершин, на таком же компьютере потребовало бы более двух лет.
- Понятно, что исследователи стали искать новые идеи позволявшие применять силовые алгоритмы для построения изображений больших графов.