Ответы на вопросы по курсу «Основы Кибернетики»

1. Определение Дизъюнктивных нормальных форм (ДНФ). Геометрическая интерпретация ДНФ. Совершенная ДНФ. Сложность ДНФ, минимальные ДНФ, кратчайшие ДНФ, функция Шеннона для ДНФ.

Введем понятие степени:

x^{α}

= x, если $\alpha=1$;

= x, если $\alpha=0$;

Рассмотрим конъюнкцию вида:

$x_1^{\alpha_1} * x_2^{\alpha_2} * x_3^{\alpha_3} \ ... \ x_n^{\alpha_n}$ – и назовем ее **элементарным произведением/конъюнкцией (ЭК).**

<u>Опр.</u> Дизъюнкция элементарных конъюнкций – **ДНФ**.

<u>Опр.</u> Гиперкуб размерности n называют множество наборов длины n наборов из *0* и *1*.

Существует 2^n наборов вида $< \alpha_1, \alpha_2, ... \alpha_n >$. Поставим в соответствие каждой конъюнкции (*) номер набора i (i = 0.. $2^n – 1$) и образуем дизъюнкцию всех конъюнкций:

$\vee_i \ (x_1^{\alpha_1} * x_2^{\alpha_2} * x_3^{\alpha_3} \ ... \ x_n^{\alpha_n})$

Существует **Теорема**: любая функция алгебры логики (ФАЛ), зависящая от 'n' аргументов, может быть представлена в форме:

$F(x_1, x_2, ... x_i ... x_n) = \vee x_1^{\alpha_1} * x_2^{\alpha_2} ... x_i^{\alpha_i} \ F(\alpha_1, \alpha_2, ... \alpha_i, x_{i+1}, ... x_n)$

Из этой теоремы вытекает ряд важных следствий:

1. Она дает возможность перейти от табличного задания функции к аналитической форме и сделать обратный переход.
2. Устанавливает так называемую функциональную полноту связок (базиса) "$\vee$, $\wedge$, -", т.к. позволит построить в этом базисе произвольную ФАЛ от произвольного числа аргументов.

Примечание:

1. Если $i \neq n$, то соответствующая форма функции называется **дизъюнктивной нормальной формой** (**ДНФ**).
2. Если i=n, то каноническая форма функции носит название **совершенной ДНФ** (**СДНФ**). Дизъюнкции берутся по тем наборам, на которых функция **f(X1,X2,...,Xn)=1**

Комментарий:

*Аналогичная теорема справедлива и для представления функции в **конъюнктивной нормальной форме (КНФ)**:*

$f(x_1, x_2, ..., x_n) = \& (x_1^{\alpha_1} \vee x_2^{\alpha_2} \vee ... \vee x_i^{\alpha_i}) \ f(\alpha_1, \alpha_2, ... \alpha_i, x_{i+1} ... x_n)$

*или при представлении в совершенной **КНФ (СКНФ)**:*

$f(x_1, x_2, ..., x_n) = \& (x_1^{\alpha_1} \vee x_2^{\alpha_2} \vee x_3^{\alpha_3} \vee \ ... \ \vee x_n^{\alpha_n})$

где: & означает, что конъюнкции берется по тем наборам, на которых $f(x_1, x_2, ... x_n)=0$.

<u>Переход от табличной формы функции к СДНФ или правило записи функции по единицам:</u>

X_1	X_2	$f(X_1, X_2)$
0	0	0
0	1	1
1	0	1
1	1	1

1. *Выбрать те наборы аргументов, на которых $f(X_1, X_2, ... X_n)=1$.*
2. *Выписать все конъюнкции для этих наборов. Если при этом X_i имеет значение '1', то этот множитель пишется в прямом виде, если '0', то с отрицанием.*
3. *Все конъюнктивные члены соединить знаком дизъюнкции $\vee$.*

Пример:

$f(X_1, X_2) = X_1X_2 \vee X_1X_2 \vee X_1X_2$

<u>Правило перехода от табличной формы задания функции к СКНФ или правило записи функции по нулям.</u>

X_1	X_2	$f(X_1, X_2)$
0	0	1

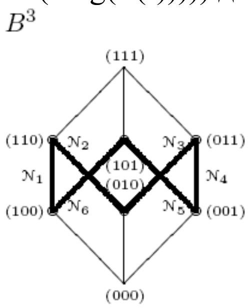
1. *Выбрать те наборы аргументов, на которых $f(X_1, X_2, ... X_n)=0$.*
2. *Если при этом X_i имеет значение '0', то остается без изменений. Если '1', то с*

0	*1*	*0*	*отрицанием.*
1	*0*	*1*	3. *Все дизъюнктивные члены соединить знаком конъюнкции* $\wedge$.
1	*1*	*0*	***Пример:*** $f(x_1,\ x_2) = (x_1 \vee x_2)\ (\ x_1 \vee x_2\ )$

<u>Опр.</u> Сложность ДНФ – число букв (литералов). **Длина ДНФ** – число элем-х конъюнкций.

<u>Опр.</u> ДНФ называется **минимальной ДНФ** если в ней минимальное кол-во букв, те минимальная сложность. При минимизации ДНФ стремятся получит форму в которой меньше букв чем в исходной, за счет построения таких элементарных произведений, которые своими единицами покрывают не одну единицу исходной функции, а несколько. По отношению к исходной ее называют **сокращенной ДНФ. Кратчайшая ДНФ** – имеющая минимальное число элем-х конъюнкций, те минимальную длину для заданной функции.

<u>Опр</u>. Функция Шеннона для ДНФ – пусть Φ - множество всех ФАЛ, а Σ(f) - множество всех схем реализующих f ∈ Φ, тогда L(n) = Max(rang(Schema(f) : на Schema(f) достигается min(rang(Σ(f))))) для всевозможных f с n переменными.



Геометрическая интерпретация: Рассмотрим наборы длины n (пусть n = 3 для простоты) из {0, 1}. Тогда можно изобразить эти наборы на n-мерном кубе. Все вершины куба можно занумеровать, так как показано на рисунке. Теперь рассмотрим набор $\gamma=(\gamma 1,\gamma 2,...,\gamma n)$ на множестве {0, 1, 2}. Пусть $\Gamma\gamma$ это множество таких вершин $\alpha=(\alpha 1,\alpha 2,...,\alpha n)$ (каждая вершина – это набор длины n из {0, 1}), где $\alpha i=\gamma i$, для всех i для которых $\gamma i\neq 2$. Число (n-r) равное числу «двоек» в γ называют размерностью грани, а число r рангом грани.

2. Тупиковая ДНФ. Сокращенная ДНФ и метод ее построения.

<u>Опр.</u> Конституентой единицы функции называют функцию, принимающую значение единицы только на одном наборе аргументов. Обычно конституенты единицы выражают через произведение всех переменных, от которых зависит функция. СДНФ – дизъюнкция конституент единицы.

<u>Опр.</u> Ранг произведения – число букв, входящих в него.

<u>Опр.</u> Собственной частью называется произведение, полученное путем отбрасывания одной или нескольких переменных. Например, `X1*X2*X3*X4`, где `X1, X1*X2, X1*X2*X3` – некоторые собственные части.

<u>Опр.</u> Если функция φ равна нулю на наборах аргументов, на которых обращается в нуль функция ϕ, то говорят, что φ является **импликантой функции** ϕ (т.е. нулей у импликанты не меньше, чем у функции).

<u>Опр.</u> Простой импликантой называется произведение, которое само входит в выражение функции, но никакая его собственная часть в выражение функции не входит.

<u>Опр</u>. Максимальные грани – грани соответствующие простым импликантам.

<u>Опр</u>. Ядровая точка (ЯТ) – точка содержащаяся только в одной максимальной грани. **Ядровая грань (ЯГ)** – грань содержащая ЯТ. **Ядро f** – савокупность всех ЯГ.

<u>Опр</u>. Сокращенной ДНФ называется дизъюнкция всех простых импликант. Ей соответсвует покрытие всеми max гранями f.

<u>Опр</u>. Тупиковой ДНФ назовем такую ДНФ что любая ДНФ полученная из нее путем удаления отдельных букв или ЭК уже не реализует функцию f. Из определения следует что тупиковая ДНФ задает тупиковое покрытие N*f* максимальными гранями *f*.

Проще это понять на множествах:

покрытие множества - набор множеств, который в объединения содержит в себе указанное множество

сокр. ДНФ - это покрытие множества, каждая компонента которого не содержит целиком другую компоненту.

тупиковая ДНФ - покрытие, не содержащие подпокрытий, т.е. любое объединение компонент , не содержит в себе компоненты, не вошедшие в объединение.
Пример: Пусть есть *f(x1, x2, x3) f = 1* на *{(001), (000), (100), (110)}* тогда сокр. ДНФ выглядит так $f = \overline{X_1}\,\overline{X_2} \cup \overline{X_2}\,\overline{X_3} \cup X_1\overline{X_3}$ а тупиковая выглядит $f = \overline{X_1}\,\overline{X_2} \cup X_1\overline{X_3}$ те тупиковая ДНФ является сокращенной, вот сокр. не всегда является тупиковой.

<u>Теорема Квайна:</u>
Если в СДНФ в начале произвести все операции неполного склеивания, а затем все операции поглощения, то в результате получится сокращенная ДНФ.

<u>Опр</u>. ДНФ Квайна – получается из сокращенной ДНФ удалением неядровых конъюнкций для которых соотв им грани покрываются ядром.

Способы построения сокращенных ДНФ

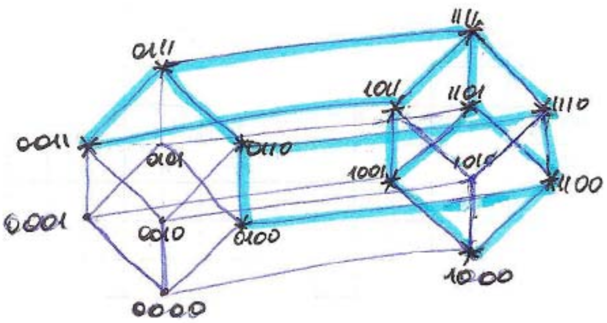
Карта Карно (для ФАЛ: n = 4)

x_1x_2 \ x_4 / x_3	0 / 0	1 / 0	1 / 1	0 / 1
0 0	1	1	1	1
0 1	1	0	1	1
1 1	1	0	0	1
1 0	1	1	1	0

Рассматриваются цилиндры и квадраты из «едениц» со сторонами равные степени 2. Выбираем по очереди цилиндры, смотрим какие переменные не меняют на его протяжении свои значения, их заносим в импликанту с отрицанием если их значение «0» и без отрицания если их значение «1». Для данной картинки сокращенная ДНФ равна

$$f = \overline{X_1}\,\overline{X_2} \cup \overline{X_2}X_4 \cup X_3\overline{X_1} \cup \overline{X_4}\,\overline{X_3} \cup X_2\overline{X_4} \cup \overline{X_1}\,\overline{X_4} \cup \overline{X_2}\,\overline{X_3}$$

Геометрическое построение



Пусть есть функция f заданная своими значениями: (0001, 1011, 1101, 1111) возьмем 4хмерный куб и те вершины на которых функция обращается в «1» пометим. После этого объедeним те вершины, которые образуют грани, и выпишем нерасширяемые грани: (--11)(-11-)(1-0-)(-1-0)(1--1)(11--) там где стоит «0» берется отрицание переменной:

$$f = x_3x_4 \cup x_2x_3 \cup x_1\overline{x_3} \cup x_2\overline{x_4} \cup x_1x_4 \cup x_1x_2$$

Через КНФ

Выписывается КНФ и раскрываются скобки используя:

<u>Тождество:</u> $X_1 * X_2 \cup X_1 * X_3 = X_1 * (X_2 \cup X_3)$

<u>Тождество:</u> $X_1 * X_2 \cup \overline{X_1} * X_3 = X_1 * X_2 \cup \overline{X_1} * X_3 \cup X_2 * X_3$

3. ДНФ типа суммы тупиковых. Критерий вхождения конъюнкции в ДНФ типа суммы тупиковых. Алгоритм построения всех тупиковых покрытий матрицы.

<u>Опр</u>. ДНФ Σ тупиковых – дизъюнкция всех тупиковых ДНФ с поледующим приведением. ДНФ Σ тупиковых получается из сокращенной ДНФ путем выбрасывания некоторых элем-х конъюнкций, до тех пор пока ДНФ не станет неприводимой.

<u>Опр</u>. Пучком *Пα(f)* назовем множество всех проходящих через α максимальных граней *f*, а точку α назовем **регулярной точкой** *f*, если найдется такая точка *β*: $\beta \in Nf$ для которой имеет место строгое включение $П\beta(f) \subset П\alpha(f)$. Грань Гγ нызовем **регулярной гранью** если все её точки регулярны.

Теорема *Простая импликанта K ФАЛ f входит в ДНФ ΣT тогда и только тогда, когда грань N_K не является регулярной гранью этой ФАЛ.*

Доказательство. Пусть $\alpha_1, \ldots, \alpha_s$ — все регулярные точки ФАЛ f. Тогда для каждого j, $j = 1, \ldots, s$, в силу регулярности точки α_j, найдется нерегулярная точка β_j ФАЛ f, обладающая тем свойством, что любая максимальная грань ФАЛ f, проходящая через точку β_j, проходит и через точку α_j. Следовательно, любая система максимальных граней ФАЛ f, покрывающая точки $\beta_1, \ldots, \beta_s$, «автоматически» покроет все точки $\alpha_1, \ldots, \alpha_s$. Таким образом, грань N_K, состоящая из регулярных точек, не может входить в тупиковое покрытие множества N_f максимальными гранями, и поэтому ЭК K не может входить в ДНФ ΣT ФАЛ f.

Пусть теперь N_K — нерегулярная грань ФАЛ f, которая содержит нерегулярную точку α, и пусть $N_f \setminus N_K = \{\beta_1, \ldots, \beta_q\}$. Из нерегулярности точки α следует, что для любого j, $j = 1, \ldots, q$, пучок $\Pi_{\beta_j}(f)$ не может быть строго вложен в пучок $\Pi_\alpha(f)$. Кроме того, равенство $\Pi_{\beta_j}(f) = \Pi_\alpha(f)$ тоже невозможно, так как $N_K \in \Pi_\alpha(f) \setminus \Pi_{\beta_j}(f)$, и поэтому в $\Pi_{\beta_j}(f)$ найдется грань N_{K_j}, которая проходит через точку β_j, но не проходит через точку α. Следовательно, из покрытия множества N_f максимальными гранями N_K, N_{K_1}, $\ldots$, N_{K_q} нельзя удалить грань N_K, так как только она покрывает в нем точку α. Таким образом, любое тупиковое покрытие множества N_f, являющееся подпокрытием указанного покрытия, будет соответствовать тупиковой ДНФ, содержащей ЭК K.

Теорема доказана.

<u>Опр</u>. Говорят что набор $\alpha \le \beta$, если выполняется $\alpha i \le \beta i$ для любого номера i. Говорят что $\alpha < \beta$ если выполняется $\alpha i \le \beta i$ и $\exists j$: $\alpha j \ne \beta j$.
<u>Опр</u>. Функция $f(\alpha)$ называется **монотонной** если $f(\alpha) \le f(\beta)$, для любых α, β: $\alpha \le \beta$. Функция $f(\alpha)$ называется **монотонной по переменной xi** если $f(\alpha) \le f(\beta)$, для любых соседних по xi наборов α, β: $\alpha \le \beta$. Из определения следует, что если функция монотонна, то она монотонна по всем своим переменным и наоборот.
<u>Утверждение</u> Если $f(\alpha)$ монотонно зависит от xi, то ни одна из ее простых импикант не может содержать отрицание xi. **<u>Следствие</u>** Монотонноая функция не содержит отрицание в своей ДНФ.

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Таблица Квайна. Строки – простые импликанты в ее сокращенной ДНФ. Столбцы соответствуют наборам на которых функция обращается в еденицу. На пересечении i строки и j столбца ставят единицу если данная импликанта обращается в еденицу на данном наборе.
Пусть функция выполняется на наборах *{100,110,010,011,001,101}* и $f = x_1\overline{x_3} \cup x_2\overline{x_3} \cup x_2\overline{x_1} \cup x_3\overline{x_1} \cup x_3\overline{x_2} \cup x_1\overline{x_2}$, тогда соответствующая таблица Квайна показана слева.

Задача для данной матрицы выделить все тупиковые подпокрытия. Покрытие строк – в каждом столбце есть хотя бы одна единица. Тупиковое покрытие – покрытие из которого ничего нельзя выкинуть.
Алгоритм построения ДНФ ΣТ: функция покрытия матрицы задается КНФ вида: $f = \bigwedge_{\text{по всем столбцам } j} \left(\bigvee_{i:\, M<i,\, j>=1} K_i \right)$, после раскрытия скобок получается ДНФ ΣТ.

4. ДНФ суммы минимальных и алгоритмические трудности ее построения.

Опр. ДНФ называется **минимальной ДНФ** если в ней минимальное кол-во букв, те минимальная сложность. **Кратчайшая ДНФ** – имеющая минимальное число элем-х конъюнкций, те минимальную длину для заданной функции.

Опр. **ДНФ суммы минимальных** – дизъюнкция всех минимальных ДНФ.

Опр. **Локальный алгоритм** – алгоритм в котором преобразование граней зависит от «состояния» «окрестности» заданного порядка.

Теорема Журавлева. ДНФ Σmin не строится в классе локальных алгоритмов.
Док-во: предположим что строится, и существует локальный алгоритм, которому на вход поступает 2х диагональная матрица Квайна, (с «1» на диагоналях) то на втором шаге алгоритм остановится т.к. не ∃ окрестности рассмотрением которой можно ограничиться.

5. Градиентный алгоритм. Оценка сложности (величины) покрытия, получаемого градиентным алгоритмом. Использование градиентного алгоритма для построения тупиковой ДНФ.

Комментарий речь идет об эвристическом алгоритме позволяющем получать «не очень длиные» ДНФ.

Градиентный алгоритм. На каждом шаге в матрице выбирается и включается в покрытие такая строка, которая покрывает наибольшее число еще не покрытых столбцов (если таких несколько из них выбирается строка с наименьшим номером). После этого те столбца которые были покрыты, удаляются из рассмотрения. Алгоритм продолжает работу пока есть столбцы – на его выходе «не очень длинная» тупиковая ДНФ.

Теорема 8.1 *Пусть для действительного γ, $0<\gamma\leqslant 1$, в каждом столбце матрицы M, $M \in B^{p,s}$, имеется не меньше, чем $\gamma\cdot p$, единиц. Тогда покрытие матрицы M, получаемое с помощью градиентного алгоритма, имеет длину не больше, чем*[1] $\left\lceil \frac{1}{\gamma}\ln^{+}(\gamma s)\right\rceil + \frac{1}{\gamma}$.

Доказательство. Пусть для построения покрытия матрицы M с помощью градиентного алгоритма потребовалось сделать q шагов, причем на шаге с номером t, $t \in [1,q]$, была выбрана строка с номером i_t. Для каждого t, $t \in [1,q)$, рассмотрим матрицу M_t, которая получается из матрицы M в результате удаления строк с номерами $\{i_1,\ldots,i_t\}$, а также покрываемых ими столбцов и которая принадлежит множеству B^{p_t,s_t}, где $p_t = p-t$ и $s_t = s\cdot\delta_t$, $0\leqslant\delta_t\leqslant 1$. Для определенности будем считать, что $M_0 = M$, $p_0 = p$, $s_0 = s$, $\delta_0 = 1$ и $p_q = p-q$, $s_q = \delta_q = 0$. Заметим, что при любом t, $t \in [0,q]$, справедливо неравенство

$$q \leqslant t + \delta_t \cdot s, \tag{8.1}$$

так как после выполнения первых t шагов алгоритма остаются не покрытыми $\delta_t \cdot s$ столбцов матрицы M, а на каждом следующем шаге покрывается не менее одного столбца.

Заметим, далее, что в каждом столбце матрицы M_t, $t \in [0, q)$, имеется не менее, чем $\gamma \cdot p$, единиц и поэтому общее число единиц в матрице M_t не меньше, чем $\gamma p s \delta_t$, а значит среднее число единиц в ее строках не меньше, чем $\gamma s \delta_t$. Отсюда вытекает, что строка матрицы M с номером i_{t+1}, которая выбирается на $(t+1)$-м шаге алгоритма и является строкой матрицы M_t с наибольшим числом единиц, содержит не меньше, чем $\gamma s \delta_t$, единиц, то есть покрывает не меньше, чем $\gamma s \delta_t$, еще не покрытых столбцов матрицы M. Таким образом, для любого t, $t \in [0, q)$, выполняются соотношения

$$s\delta_{t+1} = s_{t+1} \leqslant s_t - \gamma s \delta_t = s\delta_t (1 - \gamma)$$

из которых, с учетом $\delta_0 = 1$, следует, что

$$\delta_t \leqslant (1 - \gamma)^t \leqslant e^{-\gamma t} \tag{8.2}$$

при любом t, $t \in [0, q)$.

Выбирая значение параметра t так, что

$$t = \left\lceil \frac{1}{\gamma} \ln^+ (\gamma s) \right\rceil,$$

подставляя его в (8.1) и учитывая (8.2), получим

$$q \leqslant \left\lceil \frac{1}{\gamma} \ln^+ (\gamma s) \right\rceil + s \cdot e^{-\ln^+(\gamma s)} \leqslant \left\lceil \frac{1}{\gamma} \ln^+ (\gamma s) \right\rceil + \frac{1}{\gamma}.$$

Теорема доказана.

[1]Полагаем, что $\ln^+ x = \ln x$, если $x \geqslant 1$, и $\ln^+ x = 0$, если $0 < x < 1$.

Оценка сложности Пусть размер матриц MxN. Пусть P минимальная доля строк покрывающих столбец (среди всех столбцов). Выбрали строчку – покрыли PN столбцов (не меньше), осталось матрица (M-1)x(1-P)N. Пусть s шагов. Необходимое условие $(1-P)^s N \le 1$ откуда получаем оценку для $s \ge \log_{1-P} \frac{1}{N}$

6. Нижняя оценка функции Шеннона для СФЭ.

<u>Опр.</u> Схемой из функциональных элементов над базисом *Б* называется ориентированный (пары вершин (дуги) рассматриваются как упорядоченные) ациклический (не содержит ориентированных циклов) упорядоченный (дуги входящие в каждую его вершину пронумерованы) граф Σ, вершины которого помечены следующим образом:

- Каждый исток (вершина в которую не входит ни одна дуга) помечен некоторой БП из *X*, причем различные истоки помечены различными БП;
- Каждая отличная от истока вершина v схемы Σ помечена ФС (алфавит функциональных символов) φi где $k_i = d^+(v)$;
- Некоторые вершины Σ помечены выходными БП из *Z* так, что одной и той же вершине может быть сопоставлена одна и та же БП. При этом входные (выходные) БП, которые приписаны каким-либо вершинм Σ, считаются входными (соответственно выходными) БП Σ, а те вершины которым они сопоставлены – входами (выходами) СФЭ Σ.

Где *X* – входные БП, а *Z* – выходные БП.

<u>Опр.</u> Число ФЭ в СФЭ назовем ее сложностью.

<u>Опр.</u> Функцией Шеннона для класса СФЭ над базисом Б назовем функцию натурального аргумента n которая равна максимуму сложности по всем функциям от n переменных. Функция Шеннона характеризует сложность самой сложной ФАЛ.

<u>Лемма</u> Для любой ФАЛ от *n* перменных существует СФЭ реализующая *f* со сложностью не более чем $n*2^{n+1}$

<u>Утв</u>. Число схем с n входами и одним выходом содержащим h элементов базиса Б($\neg,\wedge,\vee$), те $S(n,1,h) \le \frac{1}{h!}3^n(n+h)^{2h+1}$

<u>Теорема</u> $L(n) > \frac{2^n}{n}$

Док-во

$$S(n,1,h) \le \frac{1}{h!}3^n(n+h)^{2h+1}$$

$$S(n,p,h) = \sum_{i=1}^{h} S(n,1,i) \le \sum_{i=0}^{h} \frac{1}{i!}3^n(i+n)^{2i+1} \le (h+1)\frac{1}{h!}3^n(h+n)^{2h+1} \le \frac{1}{h!}3^n(h+n)^{2h+2}$$

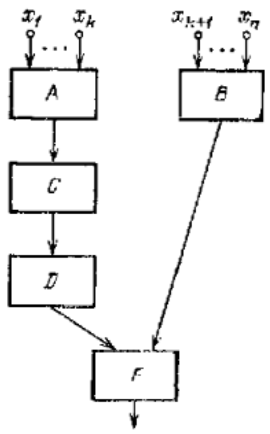
Если h=n $S(n,1,h) \le (\frac{c}{n})^h 3^h (2h)^{2h+1} = (ch)^h$, где c = const. Пусть $h = \frac{2^n}{n}$, тогда

$S(n,1,h) \le (c\frac{2^n}{n})^{\frac{2^n}{n}}$, верхняя грань кол-во функций которые реализует схема сложности h для ФАЛ от n БП. $(c\frac{2^n}{n})^{\frac{2^n}{n}} < 2^{2^n}$ при $n \to \infty$. Так что сложности $h = \frac{2^n}{n}$ не хватает для реализации всех ФАЛ, следовательно $L(n) > \frac{2^n}{n}$.

7. Ассимптотически наилучший метод О.Б.Лупанова синтеза СФЭ в базисе ($\neg,\wedge,\vee$).

Теорема для СФЭ в базисе ($\neg,\wedge,\vee$) можно построить ассимптотически наилучший метод синтеза и $L(n) \sim \frac{2^n}{n}$.

Док-во Произвольная ФАЛ задается с помощю таблицы размера $2^k 2^{n-k}$. Строки нумеруются по наборам значений первых *k* переменных, столбцы по оствшимся *n-k*. Столбец с номером $\sigma_{k+1}\sigma_{k+2}...\sigma_n$ задает функцию $F(x_1x_2...x_k\sigma_{k+1}\sigma_{k+2}...\sigma_n)$ которая в свою очередь является компонентой разложения исходной функции *f (в СДНФ)*. На пересечении строки $\sigma_1\sigma_2...\sigma_k$ и столбца $\sigma_{k+1}\sigma_{k+2}...\sigma_n$ помещаем значение *f* в этой точке.
Возбмем целое число 1 < s < 2^n и разрежим таблицу на полосы шириной s, которые занумеруем. Рассмотрим полосу с номером i. Она распадается на короткие столбцы высотой *s* видов которых не более чем 2^s, пронумеруем их. Пусть $\gamma_1\gamma_2...\gamma_s$ столбец jого сорта, обозначим за $F_{ik}(x_1x_2...x_k) = \begin{cases} \gamma_l, (\sigma_1\sigma_2...\sigma_k) = (\sigma_1(l)\sigma_2(l)...\sigma_k(l)), l = 1..s \\ 0, (\sigma_1\sigma_2...\sigma_k) \notin i\text{ой полосе} \end{cases}$



1. Блок A реализует все конъюнкции $x_1^{\sigma_1} \ldots x_n^{\sigma_k}$

$$L(A) \leqslant k2^k.$$

2. Блок B реализует все конъюнкции $x_{k+1}^{\sigma_{k+1}} \ldots x_n^{\sigma_n}$

$$L(B) \leqslant (n-k)2^{n-k}.$$

3. Блок C реализует по совершенной д. н. ф. функции $f_{ij}(x_1, \ldots, x_k)$

$$L(C) \leqslant (s-1)(t(1) + \ldots + t(p)) < sp\,2^s.$$

4. Блок D реализует функции $f(x_1, \ldots, x_k, \sigma_{k+1}, \ldots, \sigma_n)$ на основе формулы (2)

$$L(D) \leqslant (p-1)2^{n-k} < p2^{n-k}.$$

5. Блок F реализует функцию $f(x_1, \ldots, x_n)$ исходя из ее разложения (1)

$$L(F) \leqslant 2^{n-k} + 2^{n-k} - 1 < 2 \cdot 2^{n-k}.$$

Суммируя полученные оценки, имеем

$$L(\Sigma) \leqslant k2^k + (n-k)2^{n-k} + sp2^s + p2^{n-k} + 2 \cdot 2^{n-k}.$$

Положим $k = [3\log_2 n]$, $s = [n - 5\log_2 n]$. Тогда

$$p = \left]\frac{2^k}{s}\right[\sim \frac{2^k}{s}$$

и

$$L(\Sigma) \leqslant k2^k + n2^{n-k} + 2^{k+s} + \frac{2^n}{s},$$

причем

$$k2^k = o\left(\frac{2^n}{n}\right),$$

$$n2^{n-k} = o\left(\frac{2^n}{n}\right),$$

$$2^{k+s} = o\left(\frac{2^n}{n}\right),$$

$$\frac{2^n}{s} \sim \frac{2^n}{n}.$$

Поэтому

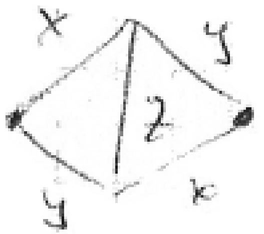
$$L(\Sigma) \lesssim \frac{2^n}{n}.$$

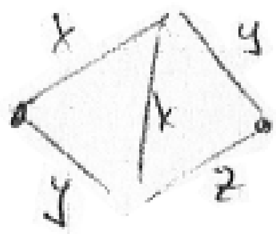
В силу произвольности функции $f(x_1, \ldots, x_n)$ отсюда следует, что

$$L(n) \lesssim \frac{2^n}{n}.$$

8. Контактные схемы. Схемы для медианы. Схемы Кардо.

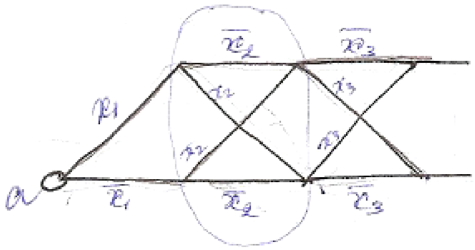
<u>**Опр.**</u> **Контактная схема** – не ориентированный граф. Ребрам приписываем литералы. Если ребру приписан x то говорят что ребро проводит при x = 1 и не проводит при x = 0. Ребра называются **контактами**. Если ребро помечено литералом **без** отрицания то оно называется **замыкающим контактом**, с отрицанием – **размыкающим**. В схеме выделяют 2 и более вершины полюсы. Они образуют сеть.
<u>**Опр.**</u> **Сложность схемы** – число контактов.

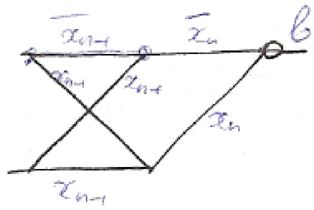




Опр. Медиана, реализует функцию:
F = xy∨yz∨zx

Опр. Схема Кардо:





Сумма по модулю «2», логический смысл чтобы оказаться наверху надо перепрыгивать по единице. Сложность: 4n-4

9. Нижняя оценка порядка функции Шеннона для контактных схем.

$L(n) = \max\limits_{f \in P_2^n} \min\limits_{\Sigma\, реализ f} (L(\Sigma))$

Как оценить снизу функцию Шеннона для КС: $L^{KC}(n)$?

Введем функионал S(L, n) – число схем от n БП сложностью не выше L.

$S(L,n) \le LL^2(2nL^2)^L$ и должно выполнятся $S(L,n) \ge 2^{2^n}$ (иначе схем не хватит)

Логарифмируем и поучаем оценку: $3\log L + L(\log 2n + 2\log L) \ge 2^n$

Утв. Неравенство нарушается если: $L \sim< \frac{2^n}{n}(\frac{1}{2} - \varepsilon)$ при $n \to \infty$

Комментарий $S(L,n) \le LL^2(2nL^2)^L$

1. первая L собственно сложность
2. Число вершин не больше чем на L больше числа контактов если схема связная – отсюда L^2
3. $2nL^2$ - варианты выбора контактов.
4. степень L так как L раз ставим 2 контакта в схему.

10. Метод Шеннона для синтеза контактных схем.

Утв для любого n можно построить универсальный многополюсник Un (схема реализующая все ФАЛ от n-1 переменной). L(Un) <= $2 * 2^{2^n}$

Теорема Шеннона Существует алгоритм который позволяет построить схему со сложностью L(n) <= $8 * \frac{2^n}{n}$ для любой ФАЛ от n переменных.

11. Метод каскадов для КС и СФЭ

$f(x_1,\dots,x_n) = \bigvee_{(\sigma_1,\dots,\sigma_k)} x_1^{\sigma_1} \& \dots \& x_k^{\sigma_k} \& f(\sigma_1,\dots,\sigma_k, x_{k+1},\dots,x_n)$

(можно не обяз. по первым k переменным, а по $\forall k$)

$f(x_1,\dots,x_n) = \bar{x}_1 \& f(0,x_2,\dots,x_n) \vee x_1 \& f(1,x_2,\dots,x_n)$

Этот метод не хуже ~~...~~ Шеннона, т.к. если на нек-ром k остановиться, то его и получим (но не лучше)

Идея основана на том, что f остаточные часто повторяются

Ф-я монот. по перем: $f(x,\dots,0,\dots) \le f(\dots,1,\dots)$

$f(x_1,\dots,x_n) = 0\, f(0,x_2,\dots,x_n) \vee x_1 f(1,x_2,\dots,x_n)$ – не сод. отриц.

по последней монотонна!!! и по x_2!!!

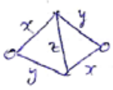
12. Самокорректирующаяся контактная схема, схемы с исправлением одного замыкания (размыкания).

Задачи: собрать схему которая будет:

- легко обнаруживать неисправности;
- работать даже при наличии неисправности;

пример: на размыкание

было

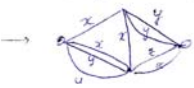


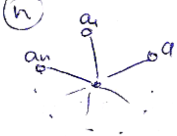
стало

на замыкание:

мин. самокоррект. схема для мостика – 8

самокор. схема для

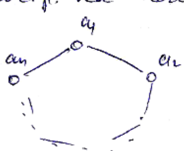




хотим, чтобы между ними всеми была проводимость x

с самокоррекцией на размык.

без коррекции – $(n-1)$ (любое дерево, постр. на полюсах)



самокоррекция на замык.

метёлка

Теорема Для любой ФАЛ найдется схема Sn порядка сложности не более $\frac{2^n}{n}$ и почти все контакты расположенные в однородных метелках размера $\Theta(\sqrt{n})$

Следствие При одной ошибке сложность ассимптотически не увеличивается.

13. Асимптотически наилучший метод синтеза СФЭ корректирующих неисправность фиксированного числа ненадежных элементов.

14. Эквивалентные преобразования формул в базисе ∨,&,¬,0,1,x.

Опр. Формулы Ф1 и Ф2 эквивалентны если задают равны е функции.
Эквивалентность можно определить перебором значений, выписав СДНФ,
Рассмотрим (∨,&,¬,0,1,x):

1. Коммутативность x∨y = y∨x; x&y = y&x;
2. Ассоциативность x∨(y∨z) = (x∨y)∨z; x&(y&z) = (x&y)&z;
3. 0 = ¬x&x; 1 = ¬x∨x;
4. ¬(x&y) = ¬x∨¬y; ¬(x∨y)= ¬x&¬y;
5. x&(y∨z) = x&y∨x&z; x∨(y&z) = (x∨y)&(x∨z);

15. Теорема перехода

Сокращение конечной полной системы тождеств (**КПСТ**)

Наличие конечной полной системы множеств не зависит от базиса для замкнутого класса.

Смысл: B_1, B_2 – базисы,
КПСТ для B_1: $A_1 \sim A_1', \ldots, A_s \sim A_s'$ (P_1 … P_s ф-ла)
=> для B_2 ∃ базис КПСТ $C_1 \sim C_1', \ldots, C_2 \sim C_2'$
причём это делать конструктивно.

Док-во: $\Phi_1 \sim \Phi_2 \sim \ldots \sim \Phi_{n-1} \sim \Phi_n$ если $\Phi_1 \sim \Phi_n$
и есть $G_1 \sim G_m$ в B_2
Надо найти преобр, переводящие друг в друга
В $P_1, \ldots, P_s$ можно заменить все ф-ции в них на ф-ции базиса B_2 и получим экв. преобр. в базисе B_2

16. Пример Линдона

есть ли класс, пород. конечной системы ф-ций,
но нет конечной полной системы тождеств

Док-во отсутствия: схема:

1) пример объекта
2) пример ∞ полной сист. тожд.
3) конечная сист. тожд. обладает некими св-вом, к-рым не обл. нек. тождество

Пример Линдона:
ф-я от 2ух перем из семиэл. логики

Таблица

	2	3	4
1	1	5	6
5	5	5	5
6	6	6	6

остальные - нули

система - замыкание этой ф-ции

Св-ва этой ф-ции:

от равных перем. $\equiv 0$.

$x\cdot x = 0$

$x\cdot(\ \ ) = 0$.

т.е. если второй аргумент - не переменная.

Линдон

$f(x_1, x_2) =$

$x_1 \backslash x_2$	2	3	4
1	1	5	6
5	5	5	5
6	6	6	6

Св-ва: $x\cdot x = 0$

$x\cdot(\dots\dots) = 0$ ← выражение

$0\cdot x = x\cdot 0 = 0$

Какие вообще могут быть отличные от 0 ф-ции?

$(\dots((x_1\cdot x_2)\cdot x_3)\dots)\cdot x_n$, $\quad x_2,\dots,x_n \in \{2,3,4\}$, $x_1 \in \{1,5,6\}$

У этой ф-ции все переменные существенные (если одна $=0$, то все зануляется)

Что будет, если x_2 в конце приписать,

$((\dots((x_1\cdot x_2)\cdot x_3)\cdot\dots)\cdot x_n)\cdot x_2$

если $x_1=5$ $\quad f=5$

$x_1=6$ $\quad f=6$

$x_1=1$ $\quad x_2=3 \Rightarrow f=5$

$x_2=4 \Rightarrow f=6$

$x_2=2 \Rightarrow$ пошла карусель

$((\dots((x_1\cdot x_2)\cdot x_3)\dots)x_n)x_1 \equiv 0$.

Обозначим $0 = x\cdot x$.

Преобразования:

A_1 $\quad 0\cdot x = 0$

A_2 $\quad x\cdot 0 = 0$

A_3 $\quad x\cdot(y\cdot z) = 0$

B_m: $((\dots((x_1 x_2)\cdot x_3)\dots)x_m)x_1 = 0 \quad m=1,2,\dots$

C_m: $((\dots((x_1 x_2)\cdot x_3)\dots)x_m)x_2 = (\dots((x_1 x_2)x_3)\dots)x_m \quad m=2,3,\dots$

Если есть бесконечная система тождеств, у кот. нельзя выделить конечную, то конечной $\nexists$.

Теперь докажем, что A, B, C – полная система

Для этого предложим кан. вид для всех формул

если не 0, то переменные встречаются 1 раз

Итак, кан. вид:

Как вообще можно док-ть факт, что не конечная полная система тождеств?

если есть бесконечная система, у кот. нельзя выделить конечную ⇒ т.к. конечн. и бескон. одновр. независимых $\exists$ не могут ⇒ конечной нет

⇓

должны предложить каноническ. вид для всех формул:

1) $x \cdot x = 0$
$0 \cdot x = 0$ ..., (все, что было)

2) считаем, что каждая переменная встречается 1 раз (если не 0)

⇓

<u>кан. вид:</u>

или 0
или $((x_{i_1} \dots x_{i_2}) x_{i_3}) \dots) x_{i_n}$

никакая перестановка невозможна
т.о. все эти ф-ции различны и их не надо переводить друг в друга

0 всё поглощает

<u>Св-ва C^n:</u>

1. Ф-ла содержит все символы $x_1, \dots, x_n$
2. $u \cdot v$ и $u(v \cdot w)$ быть не должно (ограничения на расстановку скобок)

~~3. Ф-ла имеет вид~~

3. Ф-ла имеет вид $x_{i_1}, \dots x_{i_p}$ (скобки там расставлены) ~~их все второе~~

4. И если есть второе вхождение x_{i_1}, то между ними должны быть все остальные переменные

~~Итого~~

<u>Утв:</u> B_m с помощью всего остального не получается

У B_m л.ч. удовл. св-ву C^m

Среди св-в C_m нет тех, к-рые нарушают C^n
⇒ как бы ни крутили, никак не можем нарушить C^n

Справа не повторы, слева между x_1 нет x_2
⇒ C_m напрямую нарушает C^n

17. Проблема контроля управляющих систем. Тесты для таблиц. Тривиальные оценки. Верхняя оценка длины диагностического теста для почти всех таблиц.

Постановка 1: (диагностический тест)
предп., что ф-я ∈ некоторому классу.
Хотим определить конкретно (какая именно из класса)

Пост. 2: (проверяющий тест)
имеем эталонную ф-ю
проверяем, она там или не она

мера – кол-во испытаний, необх. для того, чтобы сделать вывод

сняв показания, должны обработать
(это уже др. задачи)

мы хотим знать, сколько испытаний

Мера задачи – кол-во наборов достаточное чтобы сделать вывод.

Подход 1:
составляем некое мн-во наборов → подаем → → выход → какая ф-я

диагностич. тесты
условные ← → безусловные

в зав-ти от рез-та предыдущ. набора выбираем след. набор

Тест для таблицы

таблица: (матрица из 0 и 1)
столбцы $f_0 \dots f_n$
строки $a_1 \dots a_m$

$$\begin{array}{c} \\ a_1 \\ \vdots \\ a_m \end{array} \begin{array}{c} f_0 \dots f_n \\ \left(\begin{array}{c} \\ \\ \\ \end{array} \right) \end{array}$$

проверяющим тестом называется мн-во строк, в кот. эталонный столбец (f_0) отличен от всех остальных

диагностический тест – мн-во строк, в кот. все столбцы различны

длина теста – кол-во строк
если матрица $m \times (n)$
⇒ длина диагностич. теста $T_D(n) \geq \lceil \log n \rceil$ $T_{пров} = n$;

Теорема Мура: $\forall$ матрицы $T_D(n) \leq n-1$.

что надо, чтобы первые l строк образовывали диагностический тест?

$$\begin{array}{c|c} & f_1 \;\ldots\ldots\; f_n \\ \hline a_1 & \\ \vdots & \\ a_l & \\ \vdots & \\ a_m & \end{array}$$

(какова вероятность того, что все n столбцов будут различны?)

вероятность 1 — $\frac{1}{2}$ ($\forall$ эл-та матрицы)

матриц всего 2^{nl}

$$\frac{2^l(2^l-1)\cdot\ldots\cdot(2^l-n+1)}{2^{nl}} \geq \left(\frac{2^l-n}{2^l}\right)^n = \left(1-\frac{n}{2^l}\right)^n \geq$$

$$\geq 1-\frac{n^2}{2^l} \geq \{l = 2\log n + \varphi(n)\} \geq 1-\frac{1}{2^{\varphi(n)}}$$

если $\varphi(n) \to \infty$,
то первые n строк
~~таких~~ почти наверняка образуют
диагностический тест.

18. Градинтный алгоритм. Алгоритм построения всех тупиковых тестов.

19. Полный диагностический тест для контактных схем.

20. Проблема NP полноты. Теорема Кука (формулировка). NP полнота языка КЛИКА.

Недетерминированная машина Тьюринга

машина Тьюринга — автомат

состоит из

1) бесконечная лента, на которой записаны символы из некоторого алфавита
(у нас — по крайней мере 3 символа: $\{\underset{\text{пустой}}{\Lambda}, 0, 1\}$

2) в каждый конкр. момент времени головка обозревает только одну ячейку

как. в некотором состоянии q_1, видит символ a →

→ переходит в сост. q_2
пишет b
переход в ячейку M.

есть завершающее состояние.

$$q_1 a \longmapsto q_2 b M$$

Зацикливание → таймер.
→ полиномиальное время.

алфавит A
язык – ~~...~~ $L \subseteq A^*$
(A^* – мн-во слов из A)

недетермин. машине разрешается иметь 2 и более команды с одинаковой левой частью

язык ∈ классу NP, когда он может быть распознан за полиномиальное время (машина скажет, что это слово ∈ языку)

$L_1 \overset{poly}{\propto} L \iff$ (полиномиальная сводимость)
$\subseteq A^*$, $\subseteq A_1^*$

1) ∃ машина Тьюринга, кот. преобразует слова из A^* в слова из A_1^* при этом далее

2) машина Тьюринга работает полиномиальное время относит. длины

3) слово $\tilde{\alpha} \in L_1 \iff \varphi(\tilde{\alpha}) \in L$
то, что перекодировала машина

Отношение полиномиальной сводимости транзитивно

L – NP полный язык, если:

1) $L \in NP$
2) $\forall L_1 \in NP \Rightarrow L_1 \overset{poly}{\propto} L$ (полиномиально сводится к L)

выполнимость КНФ:

дана КНФ $(\vee \vee \vee) \& (\vee \vee \ldots) \cdots$
вопрос: есть ли такой набор переменных, на кот. эта КНФ обращается в **1**?

ТЕОРЕМА КУКА: выполнимость NP-полна

Покажем, что КЛИКА – NP-полная задача (язык):

задача ∈ классу NP

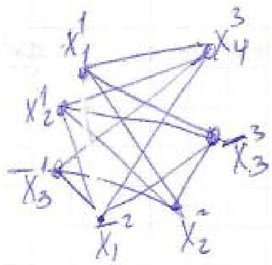
Как выполнимость сводится к Клике?
(Задачник по основам кибернетики, 2002 6 авторов)

$(x_1 \vee x_2 \vee \bar{x}_3)(\bar{x}_1 \vee x_2)(\bar{x}_3 \vee x_4)$

как по данной КНФ в соотв. с алг. $\boxed{I}$ построить граф и выдать число: КНФ выполнима $\Leftrightarrow$ в графе есть то что нужно.

т.е. надо свести зад.2 к зад.1.

Правило: строится граф, в кот. каждому литералу КНФ ставится в соотв. вершина:
все вершины соед., кроме тех, кот.
1) относятся к одной скобке
2) соотв. переменной и её отриц. ||



k = число скобок
в графе есть k-к
$\Leftrightarrow$ КНФ вып.

<u>21. NP Поднота языка 3-ВЫП и полиномиальность языка 2-ВЫП</u>

<u>Задача ~~3-КНФ~~ 3-выполнимость КНФ.</u>

<u>Формулировка 1</u>
выполнимость КНФ в каждой скобке кот. не более 3 слагаемых

выполнимость сводится к 3-вып:

$K(x_1 \vee x_2 \vee x_3 \vee x_4) \longmapsto (x_1 \vee x_2 \vee y)(\bar{y} \vee x_3 \vee x_4)K$

вып., если y не вх. в K

$(x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_5) \longmapsto (x_1 \vee x_2 \vee y_1)(\bar{y}_1 \vee x_3 \vee y_2)(\bar{y}_2 \vee x_4 \vee x_5)$

$(x_1 \vee \ldots \vee x_n) \longmapsto \underbrace{(x_1 \vee x_2 \vee y_1)(\bar{y}_1 \vee x_3 \vee y_3)\ldots(\bar{y}_{n-3} \vee x_{n-1} \vee x_n)}_{(n-2)\text{ скобки}}$

$\Downarrow$

3-КНФ - NP-полная задача

<u>2-выполнимость</u> $\in P$ (полином. задача)

напр.: $(x \vee y)(x \vee z) = (x \vee yz)$

$\downarrow$

можно сгруппировать в общем случае

$(x \vee y_{i_1} y_{i_2} y_{i_3} \dots)(\bar{x} \vee z_{j_1} z_{j_2} z_{j_3} \dots)\, K$

$(x \vee A)(\bar{x} \vee B)$

вып. $\Leftrightarrow$ $A \vee B$

$=0$ если если $y=0$ и $z=0$

вып. $\Leftrightarrow$ вып. $(y_{i_1} y_{i_2} \dots y_{i_s} \vee z_{j_1} z_{j_2} \dots z_{j_r})\, K$

свободно от x.

$\&_{k=1}^{s} \&_{l=1}^{r} (y_{i_k} \vee z_{j_l})$

0, если ... (то же самое)

задача решается с полином. время
(т.к. от одной переменной избавимся
за полином. время + переменных - полином.)

22. Задача о кратчайшем остовном дереве. Жадный алгоритм для нее.

Дерево – неориентированный связный граф без циклов

остовное дерево: подграф $G_1(V_1, E_1)$ графа $G(V, E)$, если G_1 – дерево и $V = V_1$

Алгоритм неоднозначен: если взять K_n, то $\forall$ остовное дерево будет кратчайшим.

„Градиентный алгоритм не думает о будущем"

Берем самое короткое ребро, чтобы не получить цикла, до $n-1$.

Теперь надо д-ть, что это дерево, остовное и крат.

Д-во

- То, что дерево – понятно т.к. $(n-1)$ ребро и без циклов

  Есть теорема: добавление ребра к дереву либо добавляет цикл, либо уменьшает на 1 число компонент связности.
  Т.е. после $(n-1)$ получим 1 комп. связности

- Остовное, т.к. $n-1$ ребро $\Rightarrow$ n вершин.
- Кратчайшее?

  $D_{ал}$, $D_{опт}$, $w(D_{ал}) > w(D_{опт})$

  У дерева алгоритма $D_{ал}$ $e_1, \dots, e_{n-1}$ $\quad e_{k+1} \neq e'_{k+1}$
  У оптим. дерева $D_{опт}$ $e_1, \dots, e_k, e'_{k+1}, \dots, e'_{n-1}$

  (может быть и $k=0$)

$e_{k+1} \neq e'_{k+1}$, $k \neq n-1$. (иначе наше дерево оптим)

(т.е. какие-то k - общие, а с $(k+1)$ - расхождения)

док-во идет от противного:

мы предположили, что:
дерево D_{opt} не оптимально
$\Rightarrow \exists$ оптим. дерево D'_{opt}, у к-рого совпадают первые $k \neq n-1$ ребер.

$\text{вес}(e_{k+1}) \leq \text{вес}(e'_{k+1})$

нельзя сделать так же для $e_{k+2}, \dots$ т.е. они уже берутся из разных мест (до этого не будет всё одинаково)

Рассм. набор ребер

$e_k, \dots, e_k, e_{k+1}, e'_{k+1}, \dots, e'_{n-1}$

Здесь есть цикл:
1) он обязательно содержит штрихованное ребро (т.к. до этого было дерево)
2) все $\forall$ штрихованные ребра $>$ веса $\forall$ нештрих. ребра

(будь он меньше, его бы выбрали раньше)

$e_k, \dots, e_{k+1}$ - min по весу, к-рые ...

из дерева D'_{opt} выбрасываем это штрих. ребро,
$\Rightarrow$ получим D''_{opt}, имеющий $\leq$ вес,
но имеет на 1 больше ребро.
по индукции получим, что все ребра совпадают

23. NP-полнота языка ВП. Свойство жадного алгоритма для задачи МВП.

Вершина v покрывает ребро e, если она является одним из концов ребра e.

Подмн-во $A \subseteq V$ вершин графа $G = (V, E)$ наз. вершинным покрытием, если вершины из A покрывают все ребра из E.

Задача, в кр-ых градиентный алг. срабатывает: вершинное покрытие. Есть граф G, число k дано. Можно ли подобрать в графе k вершин, покрывающих все ребра.

Это NP-полная задача
к этой задаче должна быть сведена КЛИКА:

$(V, E), k \propto (V, \bar{E}), n-k$ (в обе стороны)

КЛИКА

если в графе есть КЛИКА размера k:

если взять все вершины, кроме дополнения, то они покроют все ребра.

Задача! max КЛИКА
не явл. NP-полной (т. к. ответ — не 0,1)
~~не является~~
↓
NP-трудная задача (с её пом. можно решить NP-полную)

Рассм. задачу ~~NP~~ минимального вершинного покрытия:

алгоритм ~~приближенный~~
берем ∀ ребро и 2 вершины ⇒ выкидываем все ребра, к-рые они покрывают
и т. д.

Почему этот алгоритм хороший?

Введем ф-л

$$\frac{|F_{алг} - F_{опт}|}{|F_{опт}|} \leq 1.$$

где F - некий ф-л (здесь - кол-во вершин)

какую то из них взять обязан

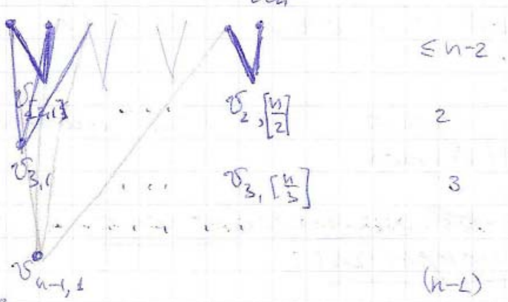
⇒ ~~увеличит~~

не больше, чем в 2 раза увеличиваем такое требуемое число вершин.

Недостатки градиентного алг:
(берем вершину, из которой макс. кол-во ребер)

есть пример: в $\ln n$ раз больше вершин, чем требуется

берем вершины $u_1, \dots, u_n$ - не соединены. Добавим:



соединена с $(n-1)$ первыми вершинами.

Что будет делать жадный алгоритм: он снизу соберет все вершины v_{ij}

Можно для покрытия взять все $u_1 \dots u_n$.

Посчитаем, какое мы получим покрытие, и во ск. раз оно хуже:

$$\left[\frac{n}{2}\right]+\left[\frac{n}{3}\right]+\ldots+\left[\frac{n}{n-1}\right] \geq \frac{n}{2}+\frac{n}{3}+\ldots+\frac{n}{n-1} =$$

$$= n\left(\frac{1}{2}+\frac{1}{3}+\ldots+\frac{1}{n-1}-1\right) = n\Big($$

$$\frac{1}{2}+\frac{1}{3}+\ldots+\frac{1}{n-1} = \sum_{j=2}^{n-1}\frac{1}{j} \geq \int_2^n \frac{dx}{x} = \ln n - \ln 2$$

верхняя
интегр. сумма

$$\Rightarrow \left[\frac{n}{2}\right]+\ldots+\left[\frac{n}{n-1}\right] \geq n\left(\ln n - \ln 2 - 1\right) \sim \underline{n\ln n}$$