ДОДАТКИ

Додаток А

Графічний матеріал кваліфікаційної роботи

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

ФАКУЛЬТЕТ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА УПРАВЛІННЯ
КАФЕДРА КІТС

Інтелектуальна система безперервного регулювання температури робочих рідин

Магістрант гр.КІТм-21-2 Сапега О.А.
Науковий керівник ст.викл. Ілюнін О.О.

ОСОБЛИВОСТІ СПОЖИВАННЯ ТЕПЛОВОЇ ЕНЕРГІЇ В ТП

2

Швидкість травління сталевої смуги – t_P нелінійно пов'язана з керованими параметрами ТП: ***T***(°K) – температурою розчину, **C** – концентрацією сірчаної кислоти в травильному розчині, та некерованою концентрацією мультігідратів сульфату заліза – C_{2n}, та тепловою енергією – **Q** :

$$\ln\frac{C(t_{n+1})_{H_2SO_4}}{C(t_n)_{H_2SO_4}} = A\cdot e^{-\left(E_a+\Delta E_T\right)/R\cdot T)\cdot\Delta t}$$

де *R*– універсальна газова константа; $C(t_{n+1})_{H_2SO_4}$ – концентрація сірчаної кислоти вкупі з C_{2n} в ванні на момент часу (t_{n+1}); $C(t_n)_{H_2SO_4}$ – теж саме на момент часу (t_n) ; A – множник Арреніуса;

$E_a = f(C,T)$ – енергія активації реакції; ΔE_T – частина теплової енергії **Q**, що передається в ТР та стабілізує швидкість травління – $t_P; \Delta t = t_{n+1} - t_n$.

Також обов'язкова умова постійного теплозабезпечення ТП величиною тепла $Q_C(t)$– умовно постійні витрати тепла:

$$\frac{dQ}{dt} \geq Q_Y(t) - Q_C(t) - Q_{RY}(t)$$

де $Q_{RY}(t)$ – *величина* потреб теплової енергії, яка *залежить від випадкової кількості окалини*, що вступає в реакцію і зменшує концентрацію (хімічну активність) ТР, та обмежена нечітким інтервалом $0 \leq Q_{RY} \leq 0.02\cdot Q_C$. В сталому ТП $Q_C \approx$ 2.893МВт, загальні витрати ТП $Q_Y \leq$3.765МВт.

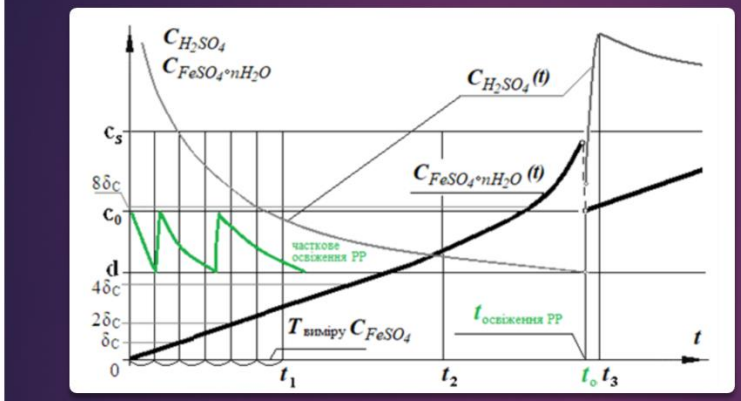
ОСОБЛИВОСТІ ВПЛИВУ НА ТП ВМІСТУ В РР МУЛЬТІГІДРАТІВ

3

Формула сполуки	Колір кристал	T_t, °C (утворення)	Y_{LR}(RGB)= *R*0.299+G*0/587+B*0.114* -оцінка яскравості	LR-інтервал RGB-коду кольору сполуки			Вплив вмісту сполуки в ТР на ТП	
				R_{L-R}	G_{L-R}	B_{L-R}		
$FeSO_4 \cdot 7H_2O$	Блакитно-зелений	-1,82÷ 56,8	119,742÷ 163,43	81÷145	147÷ 183	81÷111		Присутність в ТР $C_{FeSO_4 \circ n \cdot H_2O}$ ≤(12÷15)% зменшує t_p до $0.5 \cdot t_p^{regl}$
$FeSO_4 \cdot 4H_2O$	Зелений	56,8÷64	80,671÷85,08	0÷2	125÷ 128	64÷82		
$FeSO_4 \cdot H_2O$	Прозоро-білий	≥64	245÷255	245÷255	245÷ 255	245÷255	Останов ТП при $C_п$ ≥ 2%	

Динамічний процес травління сталі характеризується монотонною функцією росту концентрації мультігідратів за час $(0; t_o)$, яка залежить від *кількості протравленої окалини* з різною товщиною утворення та обмежена зверху величиною ГДК: при передаварійному стані $C_{FeSO_4 \circ nH_2O}(t_o) \geq C_s$ або $C_{FeSO_4 \circ 1 \cdot H_2O}(t_o) \geq 2\%$ здійснюють повну заміну РР, при $C_{H_2OSO_4}(t_o) \leq d$ – часткове освіження до регламентованої величини.

4

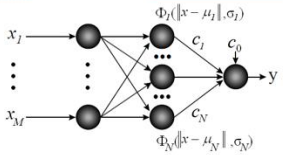


Освіження розчину при граничній концентрації шлаків

5

ІНТЕЛЕКТУАЛЬНА ІДЕНТИФІКАЦІЯ ПАРАМЕТРІВ РР

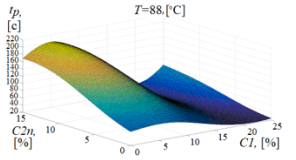
При побудові РБМ ідентифікації стану травильного розчину в якості вхідних аргументів були обрані параметри в робочих інтервалах: $C1$ – масова концентрація сірчаної кислоти H_2SO_4 (8-22%); $C2n$ – масова концентрація групи гідратів сульфатів заліза – $\sup(C2n) \le 15\%$ в ТР, **Т –** температура розчину (75-95°С). Всі параметри пов'язані нелінійно, монотонно, присутня локальна екстремальна область. Побудова гіперповерхні $t_P = f(C1, C2n, T)$ дає можливість оперативно керувати ТП в оптимальних областях параметрів.



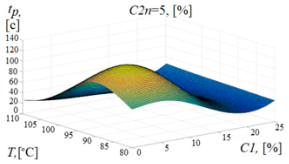
Всі запропоновані РБМ-моделі мають загальну структуру
M x N x 1

ЕКЗЕМПЛЯРИ ПОБУДОВАНИХ РБМ ПОВЕРХОНЬ $t_P = f^*(C_1, C_{2n}, T)$

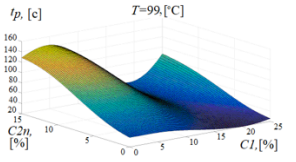
6



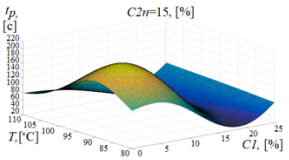
Екземпляр 3D – поверхні $t_P = f^*(C_1, C_{2n}, T = 88^\circ C)$



Екземпляр 3D – поверхні $t_P = f^*(C_1, C_{2n} = 5, T)$



Екземпляр 3D – поверхні $t_P = f^*(C_1, C_{2n}, T = 99^\circ C)$



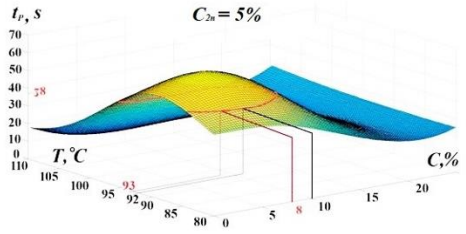
Екземпляр 3D – поверхні $t_P = f^*(C_1, C_{2n} = 15, T)$

Об'єднана гіперповерхня, з урахуванням обмежень реальних параметрів є ідентифікатором оптимальної швидкості травління – , яку визначає регламент для кожного виду сортаменту стальної смуги. Застосовано апарат RB – функцій і навчання моделі «з учителем» на вибірці вимірювань.

ВИЗНАЧЕННЯ ОПТІМАЛЬНОГО ЗАВДАННЯ T^* ДЛЯ РР

7

Транспонована доповнююча модель $T^* = f^*(C_1, C_{2n}, t_p)$, редукована за параметром C_{2n} за зрізом $t_p = regl$ оперативно визначає оптімальне завдання $T^*(k)$ при поточному значенні $C^*(k)$ в k-й момент часу відповідним регуляторам для стабілізації швидкості змотування $V(t)$.
На рис. наведено приклад для $t_P^* = 38c$ і C_{2n} = 5%.



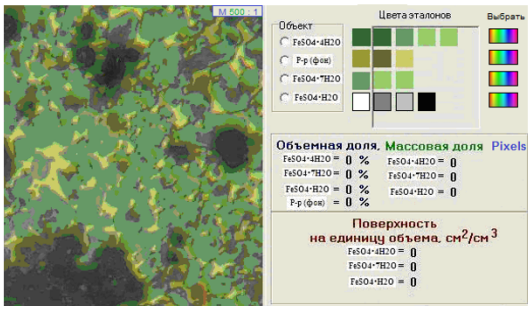
$$T^*_{(n)}(10\%, 5\%, 38c) = 92^\circ C$$

$$T^*_{(n+1)}(8\%, 5\%, 38c) = 93^\circ C$$

ІДЕНТИФІКАЦІЯ ВМІСТУ МУЛЬТІГІДРАТІВ $FeSO_4 \circ n \cdot H_2O$ В РР (1)

8

Інтерфейс системи комп'ютерного аналізу составу травильного розчину



Система, визначена нечіткою *продукційною* моделлю:

$$V_k \to C_k \ ,$$

відкрита, має можливість налаштування RGB-еталонів та донавчання у реальному часі. Під час експлуатації база даних доповнюється точковими синглетонами *(v,c)* та логічними правилами виду

$$R_i : ЯКЩО\ (V = v_i)\ ТО\ (C = c_i) \ , \text{ де}$$
$$i = 1, 2, ..., n \ .$$

Кожному зображенню ТР (файлу *.JPEG) відповідає кількісна оцінка V_k наявності k-того гідрату, що підраховується за кількістю пікселів площі фрагменту зображення: $V_k = S_k \times V_k(RGB)$, де $V_k(RGB) = \{0;1\}$ – змінна належності пікселя *(RGB)* кольору до k-го кластеру, S_k– число пікселів, що потрапляють в k-й кластер кольору. Остаточні центри кластерів установлюються з усередненими координатами: $\tau_1(1)$ = (25.33,46.67), $\tau_2(1)$ = (178, 210.67), $\tau_3(1)$ = (255.33, 313).

ІДЕНТИФІКАЦІЯ ВМІСТУ МУЛЬТІГІДРАТІВ $FeSO_4 \circ n \cdot H_2O$ В РР (2)

9

Значення C_G на виході нечіткої моделі, побудованої за допомогою FCM-процедури, об'єднує всі вихідні функції належності та інтерполює $C(V(t))$. Остаточні центри кластерів установлюються з усередненими координатами: $\tau_1(1)$ = (25.33,46.67), $\tau_2(1)$ = (178, 210.67), $\tau_3(1)$ = (255.33, 313):

$$C_G(v) = \frac{46.67 \cdot \mu_{T1}(v) + 210.67 \cdot \exp\left\{-\frac{(v-178)^2}{664.44}\right\} + 313 \cdot \exp\left\{-\frac{(v-255.33)^2}{664.44}\right\}}{\mu_{T1}(v) + \exp\left\{-\frac{(v-178)^2}{664.44}\right\} + \exp\left\{-\frac{(v-255.33)^2}{664.44}\right\}}$$

де $\mu_{T1}(v) = \exp\left\{-\frac{(25.33-v)^2}{2(50.89-42.45\omega)^2}\right\}$, а

$$\omega = \begin{cases} 1 & for \quad 0 \le v \le 25.33 \\ 0 & in \quad other \quad cases \end{cases}$$

10

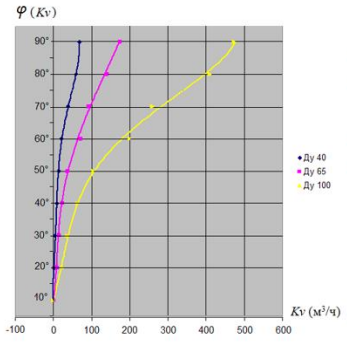
Послідовність дії моделей в ТП

Послідовність дії моделей в ТП наступна: $T^* = f^*(C_1, C_{2n}, t_p = regl) \to \varphi((G(T^*)) \to \tau(\varphi^*)$ Параметр T^* ТР формує вхідне завдання для ШНМ моделі кута поверту ДПЗ $\varphi((G(T^*))$, яка передає завдання φ^* в ШНМ обчислення тривалості керуючої напруги $G(T^*)$ для регулювання витрати $\tau(\varphi^*)$.

НЕЙРОМЕРЕЖЕВА МОДЕЛЬ КУТА ПОВОРОТУ ДПЗ

11

Дисковий поворотний затвор (ДПЗ), що регулює дебет теплоносія, має нелінійну пропускну характеристику (м^3/год), що залежить від ступеня «відкритості» ДПЗ – кута повороту . Величина забезпечує потреби витрат теплоносія для точного досягнення температури розчину .



Область визначення параметра розділяється на *n* підобластей-кластерів з центрами-медіанами, що призначаються в точках з відомими характеристиками .
При цьому і приймаються як центри сусідніх кластерів. Поточне значення витрат теплоносія відносять до *i*-го класу за критерієм мінімуму Евклідовой відстані від оцінюваного об'єкта до центру *i* -го кластера :

$$\min \quad d_{ij} = \left\| x_j - Kv_i \right\| = \sqrt{\sum_{l=1}^{n} \left(Kv_{l,i} - x_{l,j} \right)^2}$$

де $\boldsymbol{Kv_i}$ – координата центру кластера щодо осі $\boldsymbol{OX_l}$ в просторі $\boldsymbol{X_n}$; $\boldsymbol{x_j}$ – координата об'єкта щодо осі $\boldsymbol{OX_l}$ в просторі $\boldsymbol{X_n}$, в нашому випадку *n*=1.

НЕЙРОМЕРЕЖЕВА МОДЕЛЬ КУТА ПОВОРОТУ ДПЗ

12

Для кожного кластера будуються несиметричні функції належності типу:

$$\mu_{\varphi i}(G_H(t)) = \exp\left\{ -\frac{(Kv_i - G_H(t))^l}{2(\omega \cdot \delta_i' + (1-\omega)\delta_i'')^2} \right\} ,$$

де: *i*=1,2,3,…,15; формально δ_0 = 1; *l* – настроюваний показник згладжування функції;

$\omega = \{0,1\}$ – логічна змінна належності до *i*-го кластера поточного значення $G_H(t)$: $Kv_i \le G_H(t) < Kv_{i-1}$.

Для *i* = 1 область визначення $G_H(t)$ < 80, а для *i* = 15 – $G_H(t)$> 10, функція належності набирає вигляду відповідно до параметрів:

$$\mu_{\varphi 15}(G_H(t)) = \exp\left\{ -\frac{(1.57 - G_H(t))^5}{2 \cdot 0.523^5} \right\}$$

13

Параметри моделі ДПЗ DN40

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
l	2	2	2	2	2	2	3	3	3	5	5	5	5	5	5
φ_i	90	87	80	77	70	67	60	56	50	47	40	37	30	20	10
Kv_i	80	67,1	63,3	50,2	37,5	26,9	20,8	19,2	15,4	11,5	9,61	7,69	6,54	3,91	1,570
δ_i	4,3	1,27	4,37	4,23	3,53	2,03	0,53	1,27	1,3	0,63	0,64	0,38	0,88	0,78	0,523

На основі нечіткого класифікатора синтезована і налаштована модель $\varphi(G)$ структури (1-15-1). ΔT=±1°

ПІ – РЕГУЛЯТОРИ ТЕМПЕРАТУРИ РР

14

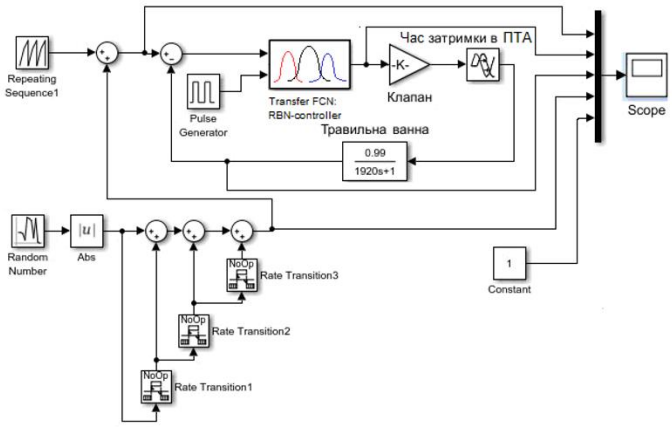
n	1	2	3	4	5	6	7
φ_n = a÷b,°	0÷30	30÷40	40÷50	50÷60	60÷70	70÷80	80÷90
k_n	0.1	0.142	0.272	0.402	0.49	0.81	1.0
T_n, c	20	6,67	6,67	6,67	6,67	6,67	6,67
P_n	159.3	99.93	54.26	33.58	31.66	19.15	15.52
I_n	0.8477	0.5128	0.2989	0.1669	0.18	0.1089	0.8822

Отримано сімейство ПІ-регуляторів для n=7 інтервалів $\varphi(G)$, визначено передавальні функції $W_R(s) = P + \frac{I}{s}$ з оптимальними настроювальними параметрами P, I регулятора, які мінімізують час виходу системи в режим. Модальний режим ТП відповідає колонці 6. де T_n відповідає позиціонуванню ДПЗ в необхідний кут φ_n, що належить *n*–му інтервалу (прийнятий коефіцієнт перерегулювання ζ_1 = 1.315).

При збільшенні числа *n* класів φ_n = a÷b до нескінченності, запропонований алгоритм в сталому режимі ТП реалізує пропорційний регулятор витрати теплоносія *Kv(t)* з установленими коефіцієнтами пропорційності P_n, і затримкою $\tau_n = (0;\ 0.6)\ c$ обумовленою часом обчислень і швидкістю ЕП. Отримані дані використані для навчання цифрового РБМ регулятор

СХЕМА МОДЕЛЮВАННЯ СИСТЕМИ З РБМ РЕГУЛЯТОРОМ *T* °C

Апріорні дані та результати модельного експерименту з сімейством ПІ-регуляторів використовувалися для навчання РБМ – регулятора за принципом «наслідуючого нейрокерування». Тривалість напруги, що подається на ЕП (SIEMENS RSM 63/8F) обчислюються РБ ШНМ (5-16-1):



$$\tau_{on}(n) = F_{NN}(\varphi(n-1), \varphi(n), \Delta\varphi(n), t_{\varphi}(n-1), t_{\varphi}(n))$$

РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ ХАРАКТЕРИСТИК РЕГУЛЯТОРІВ

16

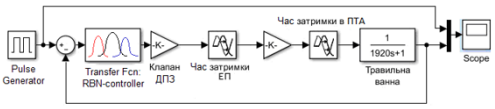


Схема РБМ моделі регулювання температури розчину

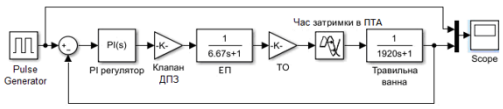
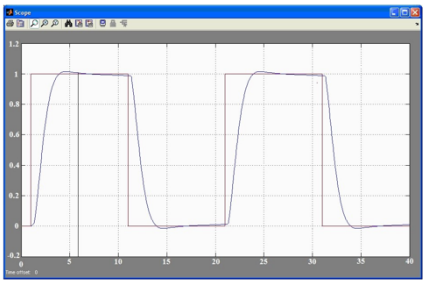
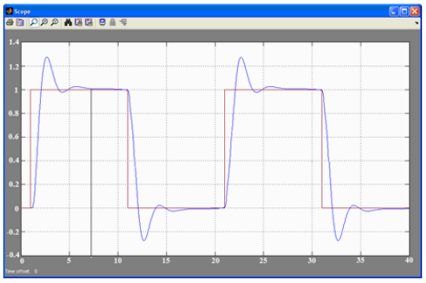


Схема моделювання ПІ-регулювання процесу



а)



б)

Результати моделювання за умови подачі на вхід послідовності прямокутних одиничних імпульсів показали: в РБМ–регуляторі (а) час входу системи в стійкий стан меньший (5.9с) в порівнянні з 6.27с у ПІ-регулятора (б); практично відсутнє перерегулювання, зменшено число коливань системи та знижена динамічна помилка.

17

Оптимальне температурне завдання

Оптимальне температурне завдання формується з огляду на вартісний критерій

Критерій керування параметрами ТП стандартної вартісної форми враховує операційні витрати:

скаляр нормованих коефіцієнтів питомих витрат *B* і ваг значущості *α* враховує зміну кожного з керованих параметрів РР **(Δ*X*)**:

ΔT – температури, ΔC – концентрації ТР, ΔC_n – концентрації солей у ТР, Δ*V* – швидкості змотування смуги.

У загальному випадку при формуванні поточних завдань регуляторам у вигляді (T^*, C^*, C_n^*, V^*) вирішується наступна оптимізаційна задача:

$$\begin{cases} \arg\min\limits_{X} t_P \\ for \quad B_1 \cdot \alpha_1 \cdot \Delta T + B_2 \cdot \alpha_2 \cdot \Delta C + B_3 \cdot \alpha_3 \cdot \Delta C_n + B_4 \cdot \alpha_4 \cdot \Delta V \to \min \end{cases}$$

18

НАУКОВА НОВИЗНА ОТРИМАНИХ РЕЗУЛЬТАТІВ

Удосконалено:

- – РБМ, що на відміну від існуючих моделей використовують гауссівські РБ функції з параметрами, налагоджуваними за методом зворотної помилки, та дозволили адекватно описати ТП безперервного травління смугової сталі в нестаціонарних умовах функціонування за наявністю невизначеностей та формувати оптимальне температурне завдання для регулювання витрат теплоносія;
- – РБМ-регулятор температури сірчаного розчину, який використовує відповідні моделі, та, на відміну від існуючих регуляторів, дозволяє адаптивно коригувати температуру розчину з урахуванням впливу змін складу розчину для підтримання його активності та регламентованої швидкості процесу за умов оптимізації енергетичних та технологічних ресурсних витрат;
- – РБМ модель, яка на відміну від існуючих реалізує нечітку кластеризацію компонентів травильного розчину, завдяки чому дозволила автоматизувати процес ідентифікації та оперативного контролю вмісту мультігідратів у розчині;

Додаток Б

```
using System;
using System.Drawing;
using System.Timers;
using System.Windows.Forms;
using System.Windows.Forms.DataVisualization.Charting;
using Stub;

namespace WindowsFormsApp1
{
    public partial class Form1 : Form
    {
        private double _countHours = 0;
        private double _countTick = 0;
        private int _countSeconds = 0;
        private double _acid = 12;
        private NetUtils _netUtils;
        private global::RadialNet.RadialNet _radialNet;
        private double _temp = 75;
        private TemperatureRadialNet _temperatureRadialNet;
        private Graphics g;
        private int _angle;
        private double _soule = 0;
        private bool _power = true;
        public Form1()
        {
            InitializeComponent();

           StartPosition = FormStartPosition.CenterScreen;
            var rn = new global::RadialNet.RadialNet(2, 9, 1,
0);
            var stub = new NetUtils();
            var tempOpt = new TemperatureRadialNet();
```

```
            _temperatureRadialNet = tempOpt;
            _netUtils = stub;
            _radialNet = rn;
            _netUtils.TrainNet(rn);
            Form1_Load();
        }

        private void Form1_Load()
        {
            timer1.Enabled = true;
            chart1.ChartAreas[0].AxisY.Maximum = 100;
            chart1.ChartAreas[0].AxisY.Minimum = 70;

            chart1.ChartAreas[0].AxisX.LabelStyle.Format =
"H:mm:ss";
            chart1.Series[0].XValueType =
ChartValueType.DateTime;

           chart1.ChartAreas[0].AxisX.Minimum =
DateTime.Now.ToOADate();
            chart1.ChartAreas[0].AxisX.Maximum =
DateTime.Now.AddSeconds(14400).ToOADate();

            chart1.ChartAreas[0].AxisX.IntervalType =
DateTimeIntervalType.Seconds;
            chart1.ChartAreas[0].AxisX.Interval = 1200;

           chart2.ChartAreas[0].AxisY.Maximum = 90;
            chart2.ChartAreas[0].AxisY.Minimum = 0;

            chart2.ChartAreas[0].AxisX.LabelStyle.Format =
"H:mm:ss";
            chart2.Series[0].XValueType =
ChartValueType.DateTime;
```

```
            chart2.ChartAreas[0].AxisX.Minimum =
DateTime.Now.ToOADate();
            chart2.ChartAreas[0].AxisX.Maximum =
DateTime.Now.AddSeconds(14400).ToOADate();

            chart2.ChartAreas[0].AxisX.IntervalType =
DateTimeIntervalType.Seconds;
            chart2.ChartAreas[0].AxisX.Interval = 1200;

           chart3.ChartAreas[0].AxisY.Maximum = 6;
            chart3.ChartAreas[0].AxisY.Minimum = 0;

            chart3.ChartAreas[0].AxisX.LabelStyle.Format =
"H:mm:ss";
            chart3.Series[0].XValueType =
ChartValueType.DateTime;

            chart3.ChartAreas[0].AxisX.Minimum =
DateTime.Now.ToOADate();
            chart3.ChartAreas[0].AxisX.Maximum =
DateTime.Now.AddSeconds(14400).ToOADate();

            chart3.ChartAreas[0].AxisX.IntervalType =
DateTimeIntervalType.Seconds;
            chart3.ChartAreas[0].AxisX.Interval = 1200;

           chart4.ChartAreas[0].AxisY.Maximum = 8;
            chart4.ChartAreas[0].AxisY.Minimum = 0;

            chart4.ChartAreas[0].AxisX.LabelStyle.Format =
"H:mm:ss";
            chart4.Series[0].XValueType =
ChartValueType.DateTime;

            chart4.ChartAreas[0].AxisX.Minimum =
```

```
DateTime.Now.ToOADate();
            chart4.ChartAreas[0].AxisX.Maximum =
DateTime.Now.AddSeconds(14400).ToOADate();

            chart4.ChartAreas[0].AxisX.IntervalType =
DateTimeIntervalType.Seconds;
            chart4.ChartAreas[0].AxisX.Interval = 1200;
        }

       protected override void OnPaint(PaintEventArgs e)
        {
            base.OnPaint(e);
            g = e.Graphics;
            SolidBrush myBrush = new SolidBrush(Color.Red);
            Pen selPen = new Pen(Color.Blue);
            g.DrawEllipse(selPen, 110, 360, 90, 90);
            g.FillEllipse(myBrush, 110 + (_angle/2), 360, 90 -
_angle, 90);
            myBrush.Dispose();
        }

        private void timer1_Elapsed(object sender,
ElapsedEventArgs e)
        {
            if (_acid <= 8)
            {
                _temp = 75;
                _acid = 12;
                _soule = 0;
                _countTick = 0;
            }
            DateTime timeNow = DateTime.Now;

            double currentTemp = _temp;
            double currentAcid = _acid;
```

```
            var res = _netUtils.GetTemperature(currentTemp,
currentAcid, _radialNet);

            var predTemp = currentTemp + res;
            var angle = _temperatureRadialNet.GetRange(_temp,
predTemp);
            var sole = _netUtils.GetSole(_countSeconds);


chart1.Series[0].Points.AddXY(timeNow.AddMinutes(10*_countHours)
.ToOADate(), predTemp);
            chart2.Series[0].Points.AddXY(timeNow.AddMinutes(10
* _countHours).ToOADate(), angle);
            chart3.Series[0].Points.AddXY(timeNow.AddMinutes(10
* _countHours).ToOADate(), _soule);
            chart4.Series[0].Points
                .AddXY(timeNow.AddMinutes(10 *
_countHours).ToOADate(), _netUtils.GetPower(angle, ref _power));

           _temp = currentTemp + res;
            _acid = currentAcid -
_netUtils.GetAcid(_countSeconds);
            _soule += sole;
            _countHours += 1;
            _countTick += 0.3 + new Random().NextDouble()/10;
            _countSeconds++;
            _angle = (int)angle;
            Refresh();

            if(_countSeconds == 26) RefreshImages();

       }

        private void RefreshImages()
```

```
        {
            chart1.ChartAreas[0].AxisX.Minimum =
DateTime.Now.AddHours(_countHours/6 - 0.3).ToOADate();
            chart1.ChartAreas[0].AxisX.Maximum =
DateTime.Now.AddHours(_countHours/6).AddSeconds(14400).ToOADate(
);

            chart1.ChartAreas[0].AxisX.IntervalType =
DateTimeIntervalType.Seconds;
            chart1.ChartAreas[0].AxisX.Interval = 1200;

           chart2.ChartAreas[0].AxisX.Minimum =
DateTime.Now.AddHours(_countHours/6 - 0.3).ToOADate();
            chart2.ChartAreas[0].AxisX.Maximum =
DateTime.Now.AddHours(_countHours/6).AddSeconds(14400).ToOADate(
);

            chart2.ChartAreas[0].AxisX.IntervalType =
DateTimeIntervalType.Seconds;
            chart2.ChartAreas[0].AxisX.Interval = 1200;

          chart3.ChartAreas[0].AxisX.Minimum =
DateTime.Now.AddHours(_countHours/6 - 0.3).ToOADate();
            chart3.ChartAreas[0].AxisX.Maximum =
DateTime.Now.AddHours(_countHours/6 -
0.3).AddSeconds(14400).ToOADate();

            chart3.ChartAreas[0].AxisX.IntervalType =
DateTimeIntervalType.Seconds;
            chart3.ChartAreas[0].AxisX.Interval = 1200;

            _countSeconds = 0;
            _power = false;
        }
    }
```

```
}
```