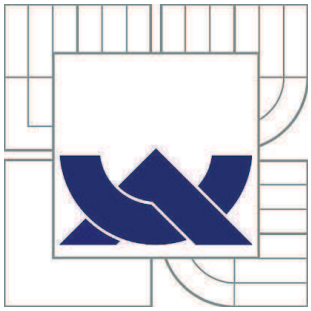


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ENERGETICKÝ ÚSTAV

FACULTY OF MECHANICAL ENGINEERING
ENERGY INSTITUTE

OPTIMALIZACE RANKINEŮVA-CLAUSIŮVA PARNÍHO CYKLU

OPTIMALIZATION OF RANKINE-CLAUSIUS STEAM CYCLE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ONDŘEJ ČEPL

VEDOUCÍ PRÁCE
SUPERVISOR

doc. Ing. JOSEF ŠTĚTINA, Ph.D.

BRNO 2013

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Energetický ústav
Akademický rok: 2012/2013

ZADÁNÍ DIPLOMOVÉ PRÁCE

student(ka): Bc. Ondřej Čepl

který/která studuje v **magisterském navazujícím studijním programu**

obor: **Energetické inženýrství (2301T035)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Optimalizace Rankineůva-Clausiůva parního cyklu

v anglickém jazyce:

Optimization Rankineůva Clausius-steam cycle

Stručná charakteristika problematiky úkolu:

Cílem je vyvinout nástroj pro numerickou optimalizaci Rankineůva-Clausiůva parního cyklu, pro různě definované účelové funkce.

Cíle diplomové práce:

Simulační model Rankinova-Clausiova cyklu v MATLABu s využitím knihovny xsteam. Optimalizovat jednotlivé parametry cyklu pro dosažení nejnižší termické účinnosti, práce turbíny a to i s uvažováním nákladů. Otestování různých optimalizačních metod.

Seznam odborné literatury:

[1] Cengel Y., Boles M.:Thermodynamics: An Engineering Approach. McGraw-Hill Professional 2010.
[2] Rao S.S.: Engineering Optimization Theory and Practice. Wiley 2009.
[3] Wagner W., Kretzschmar H.J.: International Steam Tebles. Springer 2008.
[4] Flynn D.: Thermal Power Plant Simulation and Control.The Institution of Electrical Engineers, 2003.

Vedoucí diplomové práce: doc. Ing. Josef Štětina, Ph.D.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2012/2013.

V Brně, dne 18.11.2012

L.S.

doc. Ing. Zdeněk Skála, CSc.	prof. RNDr. Miroslav Doupovec, CSc., dr. h. c.
Ředitel ústavu	Děkan fakulty

Abstrakt

Diplomová práce se zabývá optimalizací Rankine-Clausiova parního cyklu. Na začátku práce jsou popsány teoretické informace týkající se optimalizace a Rankine-Clausiova cyklu. Tyto teoretické poznatky jsou pak následně využity při tvorbě simulačních modelů v programu MATLAB. Jednotlivé simulační modely jsou varianty ideálního Rankine-Clausiova cyklu. Mezi simulační modely patří základní cyklus, nadkritický cyklu a cyklus příhřevem. Pro tyto modely jsou nalezeny ideální konfigurace a vykresleny jejich t-s diagramy.

Summary

This diploma thesis deals with an optimization of Rankine-Clausius steam cycle. The theoretical information about the Optimization and the Rankine-Clausius cycle are described at the beginning of the thesis. These theoretical findings are later used to create simulation models in MATLAB. Individual simulation models are variants of the ideal R-C cycle. Simulation models include the basic cycle model, the supercritical cycle and the cycle with reheat. For these models the optimal configurations are found and their t-s diagrams are plotted.

Klíčová slova

optimalizace, Rankine-Clausiův parní cyklus, MATLAB, simulační model

Keywords

Optimization, Rankine-Clausius cycle, MATLAB, Simulation models

ČEPL, O. *Optimalizace Rankineůva-Clausiůva parního cyklu*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2013. 59 s. Vedoucí diplomové práce doc. Ing. Josef Štětina, Ph.D..

Prohlašuji, že jsem diplomovou práci na téma „Optimalizace Rankineův-Clausiův parního cyklu“ vypracoval samostatně a uvedl jsem veškerou použitou literaturu.

Bc. Ondřej Čepl

Děkuji doc. Ing. Josefu Štětinovi, Ph.D. a RNDr. Pavlu Popelovi, Ph.D. za cenné rady, věnovaný čas a pomoc při tvorbě této diplomové práce.

Bc. Ondřej Čepl

Obsah

Úvod **3**

1 Optimalizace **4**
1.1 Lineární programování . . . 5
1.1.1 Definice lineárního programování . . . 6
1.1.2 Standardizovaný tvar lineárního programování . . . 6
1.1.3 Užití optimalizace při distribuci . . . 7
1.2 Nelineární programování . . . 9
1.2.1 Konvexní funkce . . . 9
1.2.2 Metoda vnitřního bodu . . . 11
1.3 Analýza citlivosti . . . 12

2 Rankine-Clausiův cyklus **13**
2.1 Základní ideální cyklus . . . 13
2.2 Cyklus s příhřevem . . . 14
2.3 X Steam . . . 15

3 Optimalizace Rankine-Clausiova cyklu **17**
3.1 Optimalizace ideálního cyklu . . . 17
3.2 Citlivostní analýza ideálního cyklu . . . 23
3.2.1 Závislost výsledné hodnoty účelové funkce na proměnné n_1 . . . 24
3.2.2 Závislost výsledné hodnoty účelové funkce na proměnné n_2 . . . 25
3.2.3 Závislost výsledné hodnoty účelové funkce na proměnné n_3 . . . 26
3.3 Nadkritický ideální cyklus . . . 26
3.4 Cyklus s jedním příhřevem . . . 27
3.5 R-C cyklus s více příhřevy . . . 30
3.5.1 Optimalizace velikosti příhřevů . . . 34
3.5.2 Optimalizace počtu příhřevů . . . 38
3.5.3 Ideální konfigurace . . . 39

Závěr **41**

Literatura **42**

Seznam použitých zkratek, symbolů a veličin **43**

Seznam příloh **44**

A Kód pro distribuční úlohu **45**

B Kód pro ideální cyklus **46**

C Kód pro citlivostí analýzu ideálního cyklu **49**

D Kód pro cyklus s jedním příhřevem **51**

E Kód pro cyklus s více přehřevy **54**
E.1 Kód pro optimalizaci velikosti příhřevu 57
E.2 Kód pro optimalizaci počtu příhřevů 58

Úvod

V inženýrských úlohách je třeba řešit hledání ideální konfigurace, a proto je velmi zajímavé a perspektivní využívat k tomu matematické disciplíny optimalizace. Tímto tématem se zabývá tato diplomová práce. Hlavní náplní je využití optimalizace při navrhování Rankine-Clausiova cyklu.

Práce je rozdělena na dvě části. V první části se budeme zabývat teoretickými poznatky nejen z oblasti optimalizace, ale také i z oblasti Rankine-Clausiova parního cyklu. V oblasti optimalizace se hlavně budeme věnovat definici optimalizačních úloh a pojmů s tím spjatých. Dále ukážeme, jak se z inženýrské úlohy sestavuje matematický model a posléze, jak se vyřeší. V oblasti Rankine-Clausiova parního cyklu se budeme zabývat definicí a technickou realizací tohoto cyklu a také zde budou popsány dvě varianty modifikací základního cyklu. Vývoj Rankine-Clausiova cyklu souvisí s historií parních strojů, která sahá na přelom 17. a 18. století. Hlavní rozkvět je spjatý s vynalezením parní lokomotivy. Princip parní lokomotivy je založen na žárotrubném kotli (kolem ohniště je obklopena voda), kde se generuje pára. Ta pak následně tlačí na píst, který pomocí klikového mechanizmu předává energii kolům. Podobný princip se používal i na výrobu elektřiny. V dnešní době se ale elektřina vyrábí převážně jiným způsobem. Ve vodotrubnatém kotli s bubnem (spaliny proudí spalinovodem, ve kterém je zabudováno potrubí s vodou) se generuje pára, která pak následně proudí do parní turbíny, kde se přemění energie nesená párou na rotační pohyb.

V druhé části se budeme zabývat samotnými variantami Rankine-Clausiova parního cyklu od stádia vytvoření matematického modelu až po vyhodnocení dat. V prvním příkladě budeme popisovat základní variantu ideálního cyklu a budeme se snažit nalézt jeho ideální řešení. Jako druhý příklad zvolíme citlivostní analýzu předchozí úlohy, která slouží na dokreslení problému a na zjištění podrobnější závislosti. Po citlivostní analýze zapíšeme první modifikaci, a to nadkritický oběh. Tento oběh bude zapsán z důvodu názornosti ukázky zvyšování účinnosti tímto způsobem. Poslední varianta, která bude zavedena, je příhřev. Tato varianta bude vypracována dvěma způsoby. První způsob je přidání jednoho příhřevu do základní úlohy. Toto bude sloužit pro zapsání matematického modelu a pro názornější popsání následujícího příkladu. Poslední příklad spočívá v přidání více příhřevů (není daný přesný počet) k základnímu cyklu. Cílem je snažit se o nalezení ideální konfiguraci Rankine-Clausiova cyklu s příhřevy.

1. Optimalizace

Jednou ze základních úloh konstruktéra při navrhování je hledání ideální konfigurace daného problému. Tato konfigurace může být optimální z mnoha hledisek. Z tohoto důvodu je důležité především maximalizovat pevnost soustavy, účinnost cyklu a minimalizovat hmotnost apod. Metody, jimiž zjišťujeme zmíněná hlediska, mohou být různá. Velice častým postupem pro sestavení a výpočet navrhovaného modelu je využívání předchozích zkušeností a postupů. Tento způsob je velmi užitečný a rychlý, ale nemusí vždy zaručit optimální konfiguraci. Proto se v současné době začíná používat konstrukční optimalizace, která se již tolik neopírá o dříve nabyté zkušenosti [5].

Konstrukční optimalizace vychází z matematického základu v podobě vědního oboru optimalizace, která v sobě zahrnuje více matematických disciplín např. pravděpodobnost, numeriku apod. Optimalizace spočívá v přepsání reálného modelu do matematických funkcí. Tyto funkce jsou dvojího typu - první a hlavní je takzvaná účelová funkce. Účelová funkce popisuje chování modelu a u ní hledáme body extrému, které pro nás představují výsledky. Druhým typem jsou takzvané podmínkové funkce, které slouží k omezení hledaných výsledků. Obecný přepis takto sestaveného modelu se zapisuje takto [1, 2]:

$$\begin{array}{cc} \mathbf{min} & f(\mathbf{x}) \\ \text{za podmínek} & g_i(\mathbf{x}) \leq 0 \quad pro \;\; i = 1,\; \ldots,\; m \\ & h_i(\mathbf{x}) = 0 \quad pro \;\; i = 1,\; \ldots,\; l \\ & \mathbf{x} \in \mathbf{X} \end{array} \tag{1.1}$$

kde $f,\; g_1,\; \ldots,\; g_m,\; h_1,\; \ldots, h_l$ jsou funkce definované v $\Re^n$, $\mathbf{X}$ je podmnožina $\Re^n$ a $\mathbf{x}$ je vektor o velikosti n $x_1,\; \ldots,\; x_n$. Výše uvedený problém musí být vyřešen pro hodnoty proměnných $x_1,\; \ldots,\; x_n$, které splňují omezení a jsou minimem funkce f.

Matice $\mathbf{X}$ může zahrnovat horní a dolní meze na proměnných, které i když vycházejí z dalších omezení, mohou sehrát významnou úlohu v některých algoritmech (např. nezápornost, větší než, atd.). Vektor $\mathbf{x}$ ležící v $\mathbf{X}$ splňující všechna omezení se nazývá řešení problému a množina všech takových bodů tvoří přípustnou množinu řešení. K rozlišení jednotlivých bodů přípustné množiny řešení slouží pojmy globální, lokální, vázaný a volný extrém, které je možné vysvětlit následujícím způsobem:

- Globální extrém

  Bod funkce je globální extrém, je-li nejmenší (největší) hodnotou v celém přípustném prostoru.

- Lokální extrém

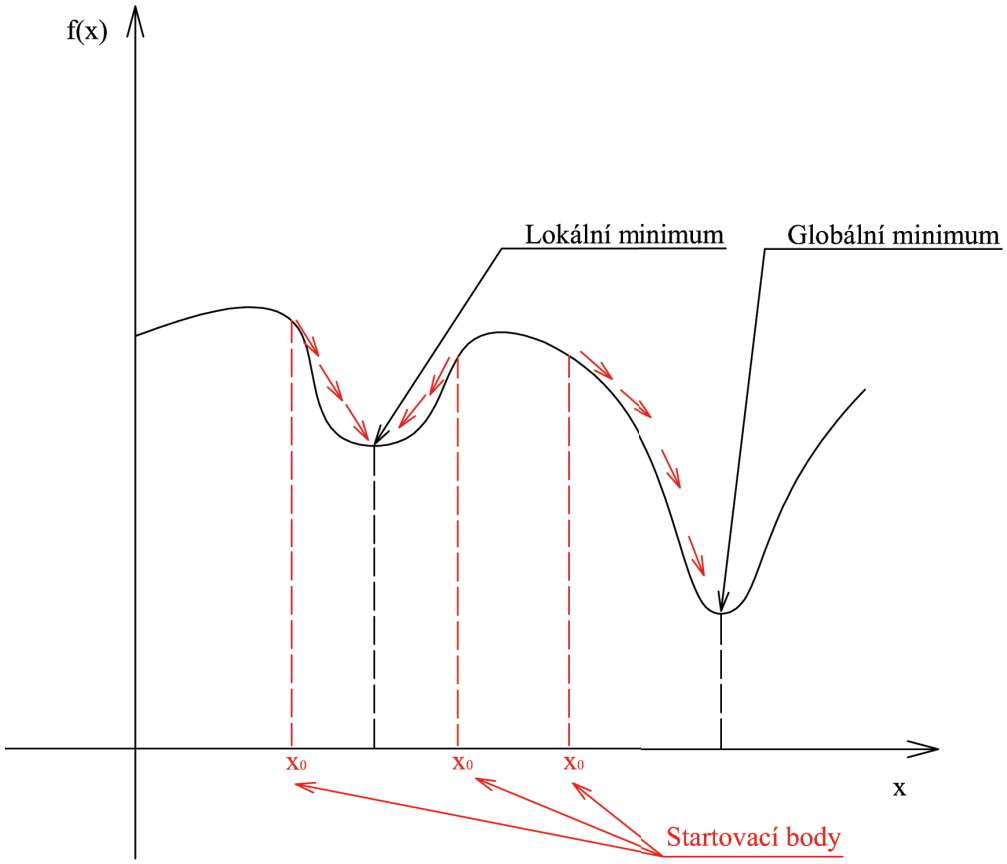
  Bod funkce je lokální extrém, je-li minimem (maximem) v části přípustného prostoru.

- Vázaný extrém

  Extrém funkce je vázaný extrém, jestliže je spjatý s podmínkovými funkcemi.

- Volný extrém

  Extrém funkce je volný extrém v případě, že není spjatý s podmínkovými funkcemi (není omezený v prostoru).



Obrázek 1.1: Problém s extrémy

Cílem konstruktéra je nalézt globální extrémy, ale to je velmi náročné. Jak je vidět na obrázku 1.1, výsledek řešení je závislý na použitém řešiči (rychlosti přibližování k bodu minima) a startovacím bodu $\mathbf{x}_0$. Tyto nedostatky se snažíme odstranit použitím více startovacích bodů při jednom řešení, ale i tak to nezaručí vždy nalezení globálního extrému.

Optimalizace se dělí na dvě základní skupiny, a to lineární a nelineární programování.

1.1. Lineární programování

Jestliže všechny funkce z 1.1 jsou lineární funkce, pak tento model je úlohou lineárního programování . Modely lineárního programování vznikají dvěma způsoby, buď po zjednodušení problému, anebo přímo z podstaty problému (jak si ukážeme v kapitole 1.1.3) [2].

Během druhé světové války, ale i po ní se ukázalo, že plánování a koordinace mezi různými projekty a efektivní využívání omezených zdrojů je nezbytné, a proto v červnu roku 1947 byl zahájen výzkum lineárního programování v US Air Force SCOOP (Scientific Computation of Optimum Programs). Výsledkem bylo, že na konci léta George B. Dantzingem vymyslel simplexovou metodu. Povědomí o lineárním programování se rychle rozšiřovalo. Už v roce 1949 se pořádala první konferenci, která byla věnována tématu

lineárního programování. Příspěvky z této konference pak roce 1951 T.C. Koopmans publikoval v knížce Activity Analysis of Production and Allocation. Tímto byly položeny základy optimalizace.

Lineární programování má mnoho výhod. Hlavní výhodou je jednoduchost a z ní následně vyplývající názornost, která nám umožňuje např. grafické znázornění výsledků, ukázku algoritmu na předem jasných výsledcích apod. Další velmi podstatnou výhodou jsou velmi rychlé a přesné algoritmy. Toto pozitivum je tak markantní, že i složité úlohy se zjednodušují, aby se převedly na lineární programování.

1.1.1. Definice lineárního programování

Jelikož jsou v modelu pouze lineární funkce, lze ho přepsat na tento tvar:

$$\begin{array}{lccccccccc}
\textbf{min} & c_1x_1 & + & c_2x_2 & + & \ldots & + & c_nx_n & & \\
\text{za podmínek} & a_{11}x_1 & + & a_{12}x_2 & + & \ldots & + & a_{1n}x_n & \geq & b_1 \\
 & a_{11}x_1 & + & a_{12}x_2 & + & \ldots & + & a_{1n}x_n & \geq & b_1 \\
 & a_{21}x_1 & + & a_{22}x_2 & + & \ldots & + & a_{2n}x_n & \geq & b_2 \\
 & \vdots & & \vdots & & & & \vdots & \vdots & \\
 & a_{m1}x_1 & + & a_{m2}x_2 & + & \ldots & + & a_{1n}x_n & \geq & b_m \\
 & x_1 & , & x_2 & , & \ldots & , & x_n & \geq & 0
\end{array} \tag{1.2}$$

kde $c_1x_1 + c_2x_2 + \ldots + c_nx_n$ je účelová funkce, kterou minimalizujeme. Hledané proměnné jsou $x_1,\ x_2,\ \ldots,\ x_n$ a $c_j,\ b_i,\ a_{i,j}$, kde $i = 1,\ 2,\ \ldots, m,\ j = 1,\ 2,\ \ldots, n$ jsou známé koeficienty.

1.1.2. Standardizovaný tvar lineárního programování

Pro jednoduchost se zavádí tzv. standardizovaný tvar lineárního programování

$$\begin{array}{lccll}
\textbf{min} & \sum_{j=1}^{n} c_jx_j & & & \\
\text{za podmínek} & \sum_{j=1}^{n} a_{ij}x_j & = & b_i & i = 1,\ \ldots,\ m \\
 & x_j & \geq & 0 & j = 1,\ \ldots,\ n
\end{array} \tag{1.3}$$

a všechny ostatní modely se na něj převádí. Převod se provádí takto:

- Maximalizace na minimalizaci

$$\textbf{max} \sum_{j=1}^{n} c_jx_j = \textbf{min} \ -\sum_{j=1}^{n} c_jx_j \tag{1.4}$$

- Nerovnice na rovnice
  Pro transformaci nerovnice na rovnice se zavádí pomocná proměnná $\mathbf{s} \geq 0$.

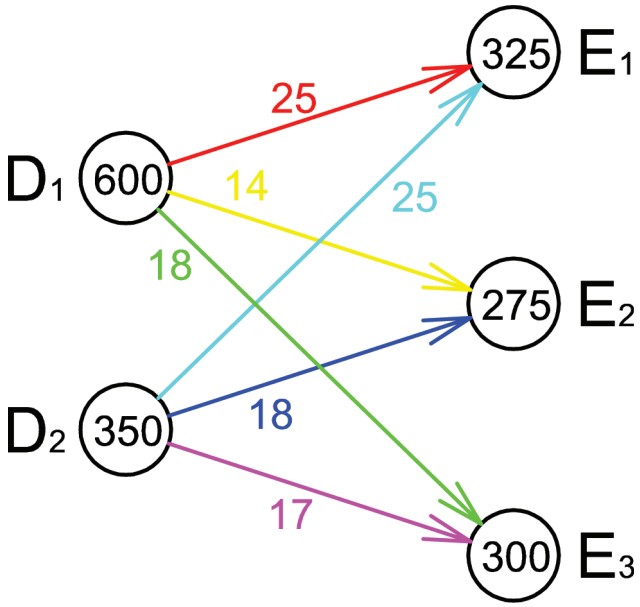
$$\sum_{j=1}^{n} a_{ij}x_j \leq b_i \Leftrightarrow \sum_{j=1}^{n} a_{ij}x_j + s_i = b_i \quad i = 1, \ \ldots, \ m \tag{1.5}$$

$$\sum_{j=1}^{n} a_{ij}x_j \geq b_i \Leftrightarrow \sum_{j=1}^{n} a_{ij}x_j - s_i = b_i \quad i = 1, \ \ldots, \ m \tag{1.6}$$

1.1.3. Užití optimalizace při distribuci

Skutečné výhody lineárního programování a jeho podstatu můžeme vidět při praktickém využití, kdy budeme optimalizovat distribuci uhlí do elektrárny. Tato úloha sice není příliš tematicky shodná s dalšími kapitolami, ale nejnázorněji představí využití lineární optimalizace v energetice.

Pro snadnější představu uvádíme následující příklad se dvěma doly uhlí D_1, D_2 s výrobními kapacitami $d_1 = 600\ kg/h$ a $d_2 = 350\ kg/h$. Uhlí dodáváme do tří elektráren E_1, E_2, E_3, které mají spotřebu $e_1 = 325\ kg/h$, $e_2 = 275\ kg/h$ a $e_3 = 300\ kg/h$. Každá přeprava uhlí z dolu do elektrárny je ohodnocená (viz obrázek 1.2). Naším úkolem je navrhnout přepravu po jednotlivých trasách tak, aby byla co nejlevnější. Ceny přepravy jsou uvedeny v (Kč h)/kg.



Obrázek 1.2: Zobrazení trasy distribuční úlohy

Pro sestavení modelu si po každé cestě označíme množství přepravovaného uhlí kladnou proměnnou. Z dolů D_1 do elektrárny E_1 proměnnou x_1 (červená trasa), z dolů D_1

do elektrárny E_2 trasu označíme x_2 (žlutá trasa), z dolů D_1 do elektrárny E_3 trasu označíme x_3 (zelená trasa), z dolů D_2 do elektrárny E_1 trasu označíme x_4 (azurovou trasu), z dolů D_1 do elektrárny E_3 trasu označíme x_5 (modrou trasu) a z dolů D_1 do elektrárny E_3 trasu označíme x_6 (fialovou trasu). Když už máme označené množství přepravy, můžeme sestrojit účelovou funkci. To provedeme tak, že pro každou trasu vynásobíme danou cenu za přepravu množstvím přepraveného uhlí a všechny tyto násobky sečteme. Jelikož je naším cílem přepravit za co nejlevnější cenu, budeme účelovou funkci minimalizovat. Matematický zápis účelové funkce je vyjádřen ve vzorci 1.7.

$$\textbf{min} \quad 25 \cdot x_1 + 14 \cdot x_2 + 18 \cdot x_3 + 25 \cdot x_4 + 18 \cdot x_5 + 17 \cdot x_6 \tag{1.7}$$

Dalším krokem, který provedeme, je sestavení podmínkových funkcí, jež jsou v tomto příkladě dvojího typu. Prvním typem jsou rovnice. Tyto rovnice zaručují dodání přesného množství uhlí do elektráren a zapisují se takto:

$$x_1 + x_4 = 325 \tag{1.8}$$
$$x_2 + x_5 = 275 \tag{1.9}$$
$$x_3 + x_6 = 300 \tag{1.10}$$

Druhým typem jsou nerovnice, a to sestavené pro výpočet maximálního výdaje z dolů. Tato hodnota nemůže být větší než výrobní kapacita. Tyto nerovnice se zapíší takto:

$$x_1 + x_2 + x_3 \leq 600 \tag{1.11}$$
$$x_4 + x_5 + x_6 \leq 350, \tag{1.12}$$

ale jelikož jsou to nerovnice, převedou se na rovnice pomocí 1.5, takže dostaneme

$$x_1 + x_2 + x_3 + s_1 + 0 \cdot s_2 = 600 \tag{1.13}$$
$$x_4 + x_5 + x_6 + 0 \cdot s_1 + s_2 = 350. \tag{1.14}$$

Z toho důvodu, že jsme přidali pro změnu nerovnic na rovnice pomocné proměnné, musíme je přidat i do účelové funkce a do předchozích omezujících funkcí, ale protože v nich se nenachází, je nutné vynásobit je 0. Celý model pak poskládáme do výsledného tvaru:

$$\begin{aligned}
\textbf{min} \quad & 25 \cdot x_1 + 14 \cdot x_2 + 18 \cdot x_3 + 25 \cdot x_4 + 18 \cdot x_5 + 17 \cdot x_6 + 0 \cdot s_1 + 0 \cdot s_2 \\
\textit{za podmínek} \quad & x_1 + x_4 + 0 \cdot s_1 + 0 \cdot s_2 = 325 \\
& x_2 + x_5 + 0 \cdot s_1 + 0 \cdot s_2 = 275 \\
& x_3 + x_6 + 0 \cdot s_1 + 0 \cdot s_2 = 300 \\
& x_1 + x_2 + x_3 + s_1 + 0 \cdot s_2 = 600 \\
& x_4 + x_5 + x_6 + 0 \cdot s_1 + s_2 = 350 \\
& x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \geq 0
\end{aligned} \tag{1.15}$$

Takto vytvořený model zapíšeme do MATLABu a pomocí řešiče *linprog* s nastavením algoritmu na simplexovou metodu to vyřešíme (viz příloha 6.1) [6]. Výsledné řešení je zapsané v tabulce číslo 1.1.

$x_1[kg/h]$	$x_2[kg/h]$	$x_3[kg/h]$	$x_4[kg/h]$	$x_5[kg/h]$	$x_6[kg/h]$	cena za přepravu [Kč]
275	275	0	50	0	300	17075

Tabulka 1.1: Výsledné řešení distribuční úlohy

1.2. Nelineární programování

Účinné a robustní algoritmy řešení lineárního programování jsou hojně používané, nicméně mnoho reálných problémů nelze dostatečně přesně zapsat nebo aproximovat jako lineární programování. Nelineární programování, jež nastává právě tehdy, když alespoň jedna funkce z modelu 1.1 je nelineární funkcí [1, 7, 3].

1.2.1. Konvexní funkce

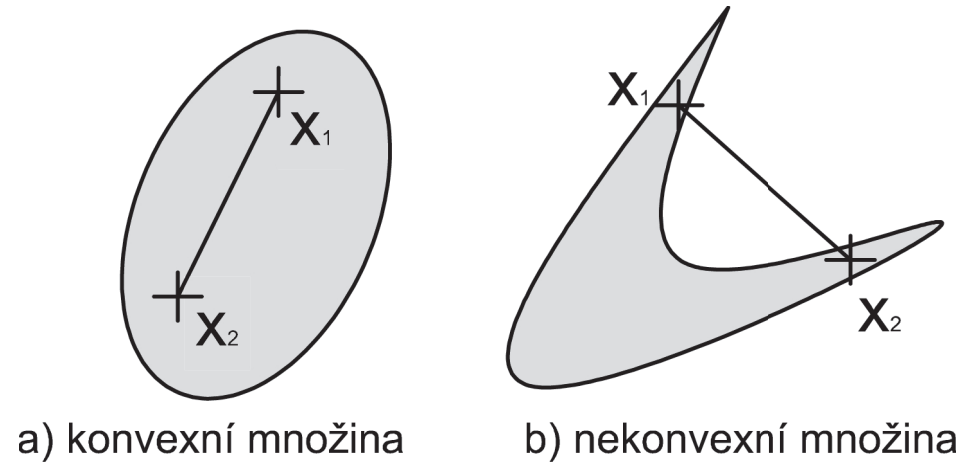
Jednou ze základních vlastností úlohy nelineárního programování je konvexnost množiny a funkce.

Konvexní množina

Množinu $S \subset \Re^n$ nazveme konvexní množinou, jestliže pro libovolné dva body $x_1,\ x_2 \in S$ a pro libovolné $\alpha \in (0; 1)$ platí

$$\alpha x_1 + (1 - \alpha)x_2 \in S.$$

Neboli v konvexní množině leží dva body a celá úsečka, která je spojuje, jak je vidět na obrázku 1.3.



Obrázek 1.3: Zobrazení konvexní a nekonvexní množiny

Konvexní funkce

Máme-li reálnou funkci $f : S \to \Re$, kde $S \subset \Re^n$ je neprázdná konvexní množina. Řekněme, že f je konvexní funkcí na S právě tehdy, když pro každé dva body $x_1,\ x_2$ z množiny S a pro libovolné $\beta \in (0; 1)$ platí

$$f(\beta x_1 + (1 - \beta)x_2) \leq \beta f(x_1) + (1 - \beta)f(x_2).$$

Pokud platí ostrá nerovnost pro každé $x_1,\ x_2$, které jsou různá, hovoříme o ryze konvexní funkci. Jestliže platí opačná nerovnost, mluvíme o konkávní (ryze konkávní) funkci.

Věta o minimu konvexní funkce

Nechť $S \subset \Re^n$ je neprázdná konvexní množina a $f : S \longrightarrow \Re$ je konvexní funkce na S. Pak je-li x_{min} lokálním minimem funkce f, potom je také bodem globálního minima funkce f. Je-li splněna i podmínka ryzí konvexnosti, je pak bod x_{min} izolované a jediné minimum.

Konvexnost množiny přípustných řešení

Označme $S_\alpha = \{\mathbf{x} \in S | f\mathbf{x} \leq \alpha\}$. Jestli je f konvexní funkce, potom S_α je konvexní množina pro všechna $\alpha \in \Re$. Této vlastnosti využijeme takto: je-li z 1.1 $\mathbf{X}$ konvexní množina a $g_i(\mathbf{x}) \leq 0$ $(i = 1,\ \ldots,\ m)$ je konvexní funkce, pak množina $C_i = \{\mathbf{x} \in \mathbf{X} | g_i(\mathbf{x}) \leq 0\}$ je konvexní množina. Z toho vyplývá, že průnik konvexních množin je konvexní množina, a tudíž množina přípustných řešení je konvexní množinou.

Spojitost

Jestliže funkce f je konvexní, pak platí, že je spojitá ve všech vnitřních bodech svého definičního oboru.

Derivace, subgradient a gradient

U konvexní funkce f není vždy zaručena existence parciální derivace (s tím související gradient), ale je vždy zaručena směrová derivace. Nechť máme konvexní množinu S a smysluplné směry $\mathbf{d} \in \Re^n$, tedy takové, že existuje $\lambda > 0$ splňující $\mathbf{x} + \lambda\mathbf{d} \in S$, pak pro všechny body $\mathbf{x} \in S$ existuje směrová derivace

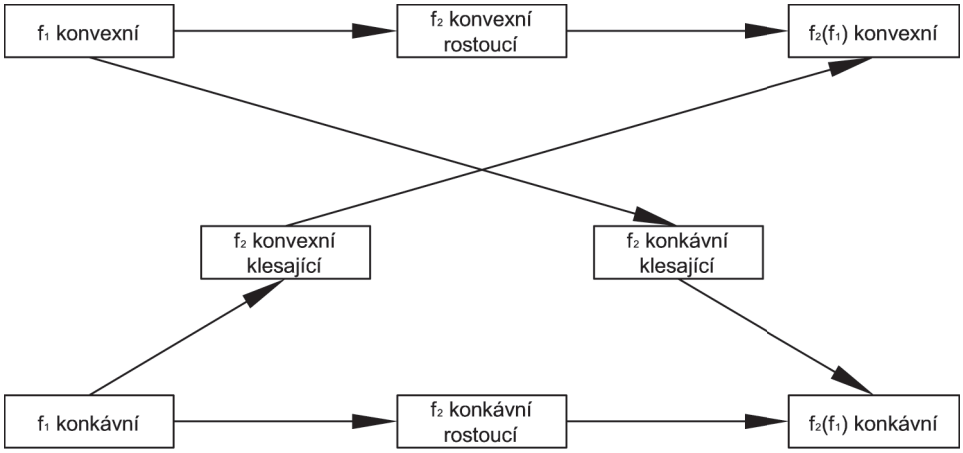
$$f_d^{'}(\mathbf{x}) = \lim_{\lambda \longrightarrow 0^+} \frac{f(\mathbf{x} + \lambda\mathbf{d}) - d(\mathbf{x})}{\lambda}.$$

Dalším pojmem je subgradient. Mějme funkci konvexní f pro niž platí, že množina bodů ležící nad grafem funkce je konvexní množinou. A pro každý vnitřní bod $\mathbf{x}_0$ množiny S existuje vektor $\mathbf{u}$ takový, že nadrovina $H = \{(x; y) | z = f(\mathbf{x}_0) + \mathbf{u}^T(\mathbf{x} - \mathbf{x}_0)\}$ je opěrnou nadrovinou množiny $epi - f$ v bodě $(\mathbf{x}_0; f(\mathbf{x}_0))$ a platí

$$f(\mathbf{x}) \geq f(\mathbf{x}_0) + \mathbf{u}^T(\mathbf{x} - \mathbf{x}_0).$$

Pak vektor u nazveme subgradient. Jeho hlavní výhodou je, že existuje ve všech vnitřních bodech množiny S, i když nemusí být jednoznačně definován.

Gradient v bodě $\mathbf{x}_0$ existuje, když je konvexní funkce f defirencovatelná v tomto bodě a platí že, je jediný gradient, který je zároveň i subgradientem. Funkce je diferencovatelná v každém bodě $\mathbf{x}_0$ otevřené S, je-li splněno $f(\mathbf{x}) \geq f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T(\mathbf{x} - \mathbf{x}_0)$ pro libovolné $x \in S$.



Obrázek 1.4: Schéma vlastností složených funkcí

Konvexnost složených funkcí

Když budeme mít reálné konvexní funkce f_1 a f_2, kde f_2 je rostoucí funkce, pak platí, že složená funkce $f_2(f_1)$ je také konvexní. Těchto vlastností pro složené funkce je více, a proto si je přehledně zapíšeme do obrázku 1.4.

Dále platí pro funkce více proměnných, že nezáporná lineární kombinace konvexních funkcí f_j

$$f(\mathbf{x}) = \sum_{i=1}^{k} \alpha_i f_i(\mathbf{x}) \quad kde\ \alpha_i > 0\ pro\ j = 1,\ \ldots,\ k$$

je konvexní funkce. A taky součet konvexních funkcí je konvexní funkce (tato podmínka nemusí platit ale pro součin).

Extrémy konvexních funkcí

Konvexní funkce f má v bodě $\mathbf{x}_0$ minimum, když v tomto bodě subgradient $\mathbf{u}$ splňuje $\mathbf{u}^T(\mathbf{x} - \mathbf{x}_0) \geq 0$ neboli neexistuje přípustný směr poklesu (nelze už hodnotu zlepšit). Je-li navíc funkce v tom bodě diferencovatelná, pak existuje jediný subgradient, který se rovná gradientu. Z toho vyplývá, že nutnou a postačující podmínkou minima v $\mathbf{x}_0$ je

$$\nabla f(\mathbf{x}_0) = \mathbf{0}.$$

1.2.2. Metoda vnitřního bodu

V MTALBu máme možnost výběru ze čtyř různých algoritmů, které umožňují řešit nelineárního programování s podmínkovými funkcemi (vypsané názvy algoritmů jsou zapsány, jak se vkládají do nastavení řešiče) [6].

1. *'trust-region-reflective'*

   Je vhodný pro velké problémy, ale pouze v případě, když jsou podmínkové funkce zastoupeny jako lineární nerovnice nebo je ohraničená oblast řešení. Ale není zaručeno, že všechny body iterací budou přípustné. Je deafaltně nastavena.

2. *'active-set'*

   Je vhodný pro obecný nelineární problém, ale též nedodržuje vždy přípustné řešení.

3. *'interior-point'*

   Je vhodný pro velké problémy s řídkou strukturou. Hlavní předností této metody je, že toleruje nadefinované omezení pro všechny body iterace.

4. *'sqp'*

   Používána pro obecné nelineární úlohy. Její předností je též, že každý bod iterace je v přípustném řešením.

Z důvodu poměrně rychlejšího výpočtu a zachování každého bodu iterace v přípustném řešením se nám nejlépe hodí metoda vnitřního bodu. Metoda vnitřního bodu (v MATLABu označovaná *interior-point*) je metoda, která je založena na bariérové funkci.

Bariérová metoda převádí úlohu nelineárního programování s omezením typu $g_i(\mathbf{x}) \geq 0$, kde $j = 1,\ 2,\ \ldots,\ m$ a $x_j \geq 0$, kde $i = 1,\ 2,\ \ldots,\ n$ na extremalizační úlohy bez podmínkových funkcích. Přípustná množina se přemění na celý prostor $\Re^n$, ale tam, kde předchozí úloha nebyla přípustná, se nastaví velká hodnota (vytvoří se bariéra). Potom nahradíme maximalizační úlohu nelineárního programování s omezením úlohou, která vyhledá volné maximum funkce [5, 7].

$$B(\mathbf{x}) = f(\mathbf{x}) - \sum_{j=1}^{m} p_j(g_j(\mathbf{x})) - \sum_{i=1}^{n} p_{m+i}(x_i),$$

kde $p_j(t)$ jsou takové funkce jedné proměnné, že

$$\lim_{t \to 0^+} p_j(t) = \infty \quad kde \quad j = 1,\ 2,\ \ldots,\ m+n.$$

Hledáme volné maximum funkce 1.16, přičemž musíme výpočet začít v bodě, který je vnitřním bodem množiny přípustných řešení původní úlohy.

1.3. Analýza citlivosti

Ve většině praktických aplikací nejsou některé z koeficientů přesně známé, a proto jsou pouze odhadovány. Rovněž koeficienty v zadané úloze se často mohou měnit a potřebujeme zjistit, jak případné změny těchto koeficientů ovlivní optimální řešení úlohy. Těmito a dalšími problémy se zabývá analýza citlivosti [2].

V našem případě se budeme zaobírat citlivostí na výsledné hodnoty účelové funkce při nadefinování jedné proměnné na přesně zadanou hodnotu. Tuto operaci budeme opakovat, a tak zjišťovat, jakou má model citlivost na danou proměnnou a především, jakou tendenci má výsledná hodnota účelové funkce na postupně měnící se hodnotu proměnné.

2. Rankine-Clausiův cyklus

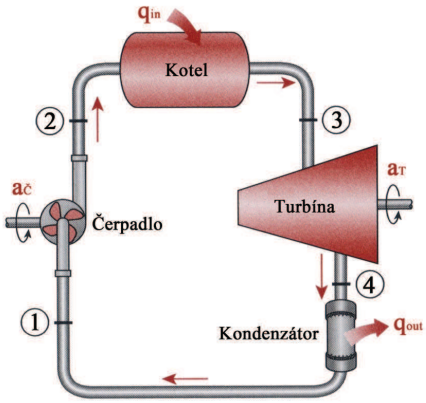
Rankine-Clausiův cyklus (dále již jen R-C cyklus) je tepelný cyklus s vodou, která během termodynamického oběhu mění svoje skupenství. Z kapalného skupenství se stává plynné (pára) a popsaný děj probíhá i opačně. Tento cyklus je hojně využíván v elektrárnách, a to v úhelných a tlakovodních jaderných [4].

Základním porovnávacím kritériem jednotlivých cyklů je tzv. termická účinnost. To je bezrozměrné číslo, které vyjadřuje, jakou má daný cyklus efektivitu v přeměně tepelné energie na užitečnou práci. Značí se η_t a pohybuje se teoreticky v intervalu 0 až 1. Velmi často se zapisuje v procentech, a to tehdy, když se vynásobí hodnotou 100. V našem případě se budeme zabývat termickou účinností R-C cyklu, jenž se obecně vyjadřuje tímto vztahem:

$$\eta_t = \frac{\textit{práce celého cyklu}}{\textit{přivedené teplo}}. \tag{2.1}$$

2.1. Základní ideální cyklus

Nyní si představíme nejjednodušší případ R-C cyklu, který se skládá z několika komponentů. Mezi jeho důležité součásti patří kotel, turbína, kondenzátor a čerpadlo, které jsou zapojeny dle schématu 2.1.

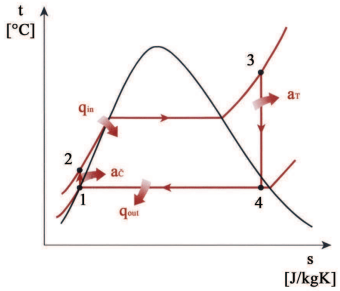


Obrázek 2.1: Schéma zapojení ideálního R-C cyklu, zdroj [4]

V tomto případě fungují jednotlivé komponenty ideálním způsobem, a proto lze napsat pro jednotlivé body schématu toto:

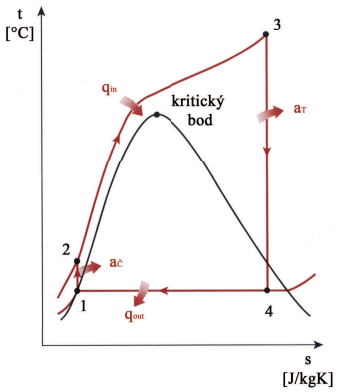
- $1 - 2$ izoentropická komprese na čerpadle, v obrázku 2.1 zapsána jako $a_{č}$,
- $2 - 3$ izobarický ohřev v kotli, v obrázku 2.1 zapsána jako q_{in},
- $3 - 4$ izoentropická expanze v turbíně, v obrázku 2.1 zapsána jako a_T,
- $4 - 1$ izobarické ochlazování v kondenzátoru, v obrázku 2.1 zapsána jako q_{out}.

Díky těmto vlastnostem můžeme zakreslit t-s diagram ideálního R-C cyklu 2.2.



Obrázek 2.2: t-s diagram ideálního R-C cyklu, zdroj [4]

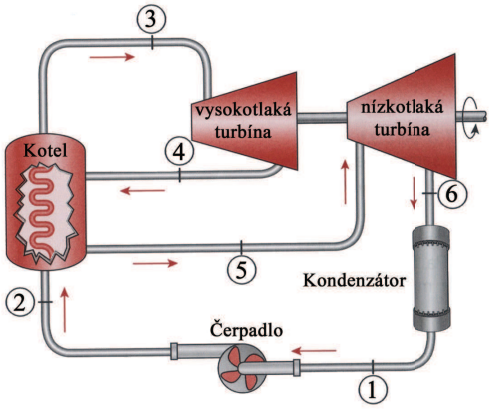
Modifikace tohoto základního cyklu může být nadkritický okruh, tedy okruh, kde povolíme překročení kritického bodu. Po této modifikaci se t-s diagram R-C cyklu zakreslí způsobem, který je znázorněn na obrázku 2.3.



Obrázek 2.3: t-s diagram nadkritického R-C cyklu, zdroj [4]

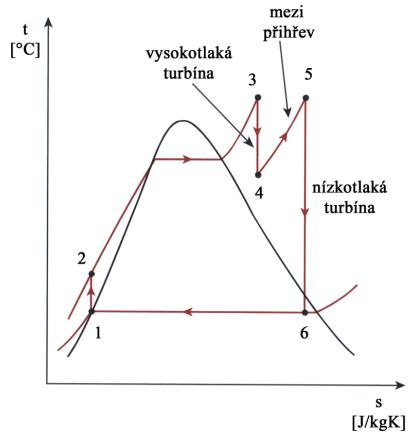
2.2. Cyklus s příhřevem

Hojně využívanou variantou je přidání takzvaného příhřevu neboli rozdělení expanze na turbíně. Mezi dvě turbíny (dvě části turbín) je umístěn přihřívač par, jak je znázorněno na schématu 2.4.



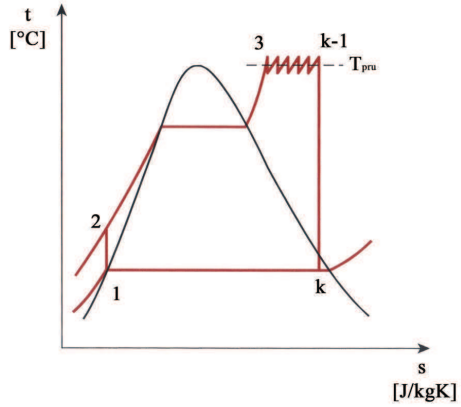
Obrázek 2.4: Schéma R-C cyklu s použitím přihřívání, zdroj [4]

Mezi body $4 - 5$ se pára znovu izobaricky ohřeje, jinak všechny ostatní děje pracují stejně jako u varianty 2.1. A proto jediné, co se změní v t-s diagramu, bude rozdělená expanze na turbíně a znázornění adiabatického přihřívání páry, jak je vidět na diagramu 2.5. Jelikož příhřev probíhá izobaricky, vstupní tlak do druhé části turbíny je nižší. Tato druhá část se nazývá nízkotlaká část a první vysokotlaká.



Obrázek 2.5: t-s diagram R-C cyklu s použitím jednoho příhřevu, zdroj [4]

Teoreticky můžeme těchto příhřevů realizovat konečně mnoho, až se přiblíží k izotermickému ději, jak je zobrazeno v 2.6.



Obrázek 2.6: t-s diagram R-C cyklu s použitím více příhřevů, zdroj [4]

2.3. X Steam

V R-C cyklu se pro přenos energie používá voda. Z tohoto důvodu budeme potřebovat její vlastnosti závislé na změně teploty a tlaku. Tento problém vyřešíme pomocí funkce *X Steam*.

X Steam pro MATLAB byl vyvinut Magnusem Holmgrenem na základě mezinárodního průmyslového standardu pro termodynamické vlastnosti vody a vodní páry, jenž se nazývá: International Association for Properties of Water and Steam Industrial Formulation 1997 (IAPWS IF-97). Uvedený standard nám poskytuje návod na velmi přesné určení vlastností vody a vodní páry v rozmezí: $0 \div 1000$ bar a $0 \div 2000$ °C. Zápis do něj je intuitivní a velmi

názorný, protože např. *XSteam('h_pt',1,20)* vrací hodnotu entalpie pro vodu o tlaku 1 bar a teplotě 20 °C [8, 9].

Jak se vidět z příkladu zápisu, program *X Steam* používá jako jednotku tlaku bar namísto pascalu, z tohoto důvodu je celý výpočet uvedený v barech. Pro úplnost si zde napíšeme převod baru na pascal:

$$1\ bar = 100\ 000\ Pa. \tag{2.2}$$

3. Optimalizace Rankine-Clausiova cyklu

V této kapitole se budeme zabývat hledáním ideální konfigurace R-C cyklu. Pro sestavení a výpočet modelu budeme využívat teoretické vlastnosti z předchozích kapitol, ke kterým přidáme vlastnosti vyplývající z technické praxe (maximálně dovolená teplota, minimální tlak apod.).

K výpočetnímu algoritmu je přidán algoritmus na kreslení t-s diagramu. Tento algoritmus je spuštěn vždy po provedení výpočtu a slouží na dokreslení a ověření správnosti výsledků. Je zapsán dvěma funkcemi, z nichž první je funkce *bodyx(ds)*. Její vstupní hodnota je *ds*, tedy přesnost vykreslení grafu (vzorkování). Tato funkce je určena pro vypočtení bodu křivky sytosti páry. Je vždy vnořená do funkce druhé, pojmenované *graf(x,ds)*, a na rozdíl od ní se nemění. Zmíněná funkce je závislá na konfiguraci problému, a proto je pro každý příklad odlišná, s výjimkou ideálního cyklu a nadkritického ideálního cyklu. Účelem funkce *graf* je napočítat body cyklu a vykreslit mezi nimi křivky odpovídajících dějů a také nakreslit křivky sytosti. Vykreslování křivek probíhá tak, že mezi jednotlivými body cyklu jsou vygenerovány pomocné body. Mezi těmito body jsou vykresleny přímky. Množství pomocných bodů závisí na vzorkování. A proto velmi záleží na tom, jaké je ds. Jestliže bude *ds* velké, výpočet jednotlivých bodů bude přesný (pomalý), tudíž i výsledný t-s diagram bude přesný a naopak.

Pro lepší pochopení převodu z matematického modelu do kódu jsou vloženy do textu vývojové diagramy. Vývojový diagram je grafický zápis jednotlivých kroků programu, a proto je to ideální varianta na popsání algoritmů, které jsou v příloze. Prvky diagramu jsou pro lepší orientaci barevné, a to následovně: start (konec) jsou růžové, běžné příkazy jsou zelené, cykly *for* jsou červené, podmínky jsou žluté, podprogramy jsou modré, vstupní data jsou okrové a spojky (prvky spojující části rozděleného diagramu) jsou hnědé.

3.1. Optimalizace ideálního cyklu

Základním příkladem optimalizace R-C cyklu je optimalizace ideálního cyklu. Tento cyklus pracuje v ideálních dějích, a proto se dá znázornit v t-s diagramu 2.2. Z tohoto diagramu vyplývá, že dodaná energie v kotli se do cyklu dodává mezi body 2 a 3, což je izobarický děj. Dodaná energie v kotli se rovná:

$$q_{in} = i_3 - i_2. \tag{3.1}$$

Energie, která z cyklu odchází na turbíně, je v diagramu znázorněna mezi body 3 a 4, což představuje izoentropický děj a výsledná hodnota se dá spočítat takto:

$$a_t = i_3 - i_4, \tag{3.2}$$

přičemž je část energie získaná na turbíně spotřebovaná na čerpadle. Tento proces je v diagramu znázorněn mezi body 1 a 2, kde se vykonává izoentropická komprese, jež lze zapsat následovně:

$$a_{č} = i_2 - i_1. \tag{3.3}$$

Tedy termická účinnost nám vychází takto

$$\eta_t = \frac{a_t - a_{\check{c}}}{q_{in}} = \frac{i_3 - i_4 - (i_2 - i_1)}{i_3 - i_2}. \tag{3.4}$$

Naším úkolem je maximalizovat termickou účinnost R-C cyklu. Z toho vyplývá, že budeme muset provést přeměnu maximalizační úlohy na minimalizační, jak je popsáno v rovnici 1.4. Dále zvolíme proměnné, kterými popíšeme celý model a budeme hledat jejich optimální hodnoty. Z podstaty problému nám vyplývají tři hodnoty. Kromě teploty jsou to také dva tlaky, na kterých cyklus pracuje. Tlaky si označíme n_3 [1] pro tlak kondenzace a n_1 pro tlak ohřívání v kotli. Teplotu v bodě tři si označíme n_2. Výsledná účelová funkce tohoto modelu je pak:

$$\textbf{min} \quad -f(n_1, n_2, n_3). \tag{3.5}$$

Takto sestavený model už by nám dal řešení, ale nemělo by adekvátní reálnou hodnotu, a proto zavedeme podmínkové funkce. Jako první budeme omezovat minimální hodnotu suchosti páry x na výstupu z turbíny (v bodě 4). Tato podmínka vyplývá z přílišného podílu obsahu syté kapaliny. Je zavedena z důvodu snížení účinnosti turbíny a regulace dalších problémů na ní např. možnost kondenzace vodní páry atd. Tedy minimální hodnotu nastavíme na $0,8$. Další podmínka je stanovena pro suchost páry na konci kondenzace (bod 1), kterou nastavíme pevně na hodnotu 1 neboli po kondenzaci zbude jen sytá kapalina. Poslední podmínky, které je nutné stanovit, jsou jen omezení pro horní a dolní meze proměnných. Pro proměnnou n_1 je dolní mez rovna 0 *bar* z důvodu nezápornosti výsledku (to platí pro všechny proměnné). Další omezení je pro horní meze proměnných $n_1 = 170$ *bar*, $n_2 = 565$ $°C$ a dolní meze $n_3 = 0,08$ *bar*, které vyplývají z konstrukčních, materiálových a cenových omezení.

Výsledný model zapíšeme, jak se zapisují hodnoty do *X Steam*. Tedy optimalizační model je takový:

$$\textbf{min} \quad \frac{i_3(n_1;n_2) - i_4(n_3;s_3(n_1;n_2)) - i_2(n_1;s_1(n_3^{'})) + i_1(n_3^{'})}{i_2(n_1;s_1(n_3^{'})) - i_3(n_1;n_2)} \tag{3.6}$$

$$\begin{aligned} \textit{za podmínek} \quad x_4(n_3; s_3(n_1; n_2)) &\geq 0,8 \\ x_1(n_3; s_1(n_3^{'}) &= 1 \\ n_1 &\leq 170 \\ n_2 &\leq 565 \\ n_3 &\geq 0,08 \\ n_1,\ n_2,\ n_3 &\geq 0. \end{aligned}$$

Když už máme takto sepsaný model, převedeme ho do *MATLABu*. Jako řešič si vybereme *fmincon*, který je určen pro úlohy nelineárního programování. Struktura zápisu do řečiče *fmincon* je následující: *fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options)*, tedy prvně zadáme účelovou funkci, pak počáteční bod algoritmu. Další čtyři parametry jsou vyhrazeny pro zápis lineárních podmínkových funkcí ($A\ x \leq b$ a $Aeq\ x = beq$). Následující dvě kolonky jsou pro dolní a horní meze hledaných proměnných, pak je pozice pro zadání

[1] Proměnné neoznačujeme x jako v kapitole 1, a to z důvodu kolize značení se suchostí páry.

nelineárních podmínkových funkcí a nakonec je to místo pro další nastavení, které využijeme pro zadání algoritmu. Jestliže nějaký z uvedených parametrů není v našem modelu zařazen (např. lineární rovnice a nerovnice), nahradí se symbolem [].

Jako algoritmus zvolíme metodu vnitřního bodu (viz kapitola 1.2.2) pro její příhodné vlastnosti. Na dovysvětlení převodu si nyní ukážeme v několika málo bodech, jak se z modelu 3.6 dostal kód viz příloha B.

1. Účelová funkce

   Účelová funkce je v našem příkladě složitější, a tak je napsaná jako samostatná funkce s jménem *ucelfunkce* a volá se tím, že se před jméno dá symbol @. Účelová funkce z modelu 3.6 je v MATLABu zapsaná takto: *f =((i3-i4-i2+i1)*100)/(i2-i3);* kde, jak je vidět, se jednotlivé entalpie počítají zvlášť, a pak se do výsledné hodnoty účelové funkce zadají. Tento nestandardní zápis je z důvodu přehlednosti a srozumitelnosti kódu.

2. Podmínkové funkce

   Podmínkové funkce jsou rovněž jako účelová funkce zapsány v samostatné funkci s názvem *omezfunkce*. Tato funkce vrací teoreticky jak soustavu nerovnic, tak i soustavu rovnic, ale v našem případě máme jen jednu nerovnici a jednu rovnici (nepočítáme horní a dolní meze, které se zapisují jinde). Rovnice musíme zapsat do požadovaného tvaru, který je obecně pro rovnice takový: $ceq(\mathbf{n}) = 0$ a pro nerovnice takový: $c(\mathbf{n}) \leq 0$. Dále pro zápis v kódu platí to, co pro účelovou funkci.

3. Horní a dolní meze

   Pro zadávání horních a dolních mezí je speciální místo, do kterého se zadává celý vektor proměnných. V našem případě je vektor dolních mezí roven nulové hodnotě, jen pro n_3 je nastaven na hodnotu $0,08$. Vektor horních mezí je také nastaven na dané hodnoty, pouze poslední proměnná ho má nastavený na nekonečno.

4. Počáteční bod

   Pro nastavení počátečního bodu se používá proměnná *n0*. Výsledek je obecně velmi závislý na počáteční hodnotě. Z tohoto důvodu je praktické zapsat výsledek optimalizace s počátečním bodem.

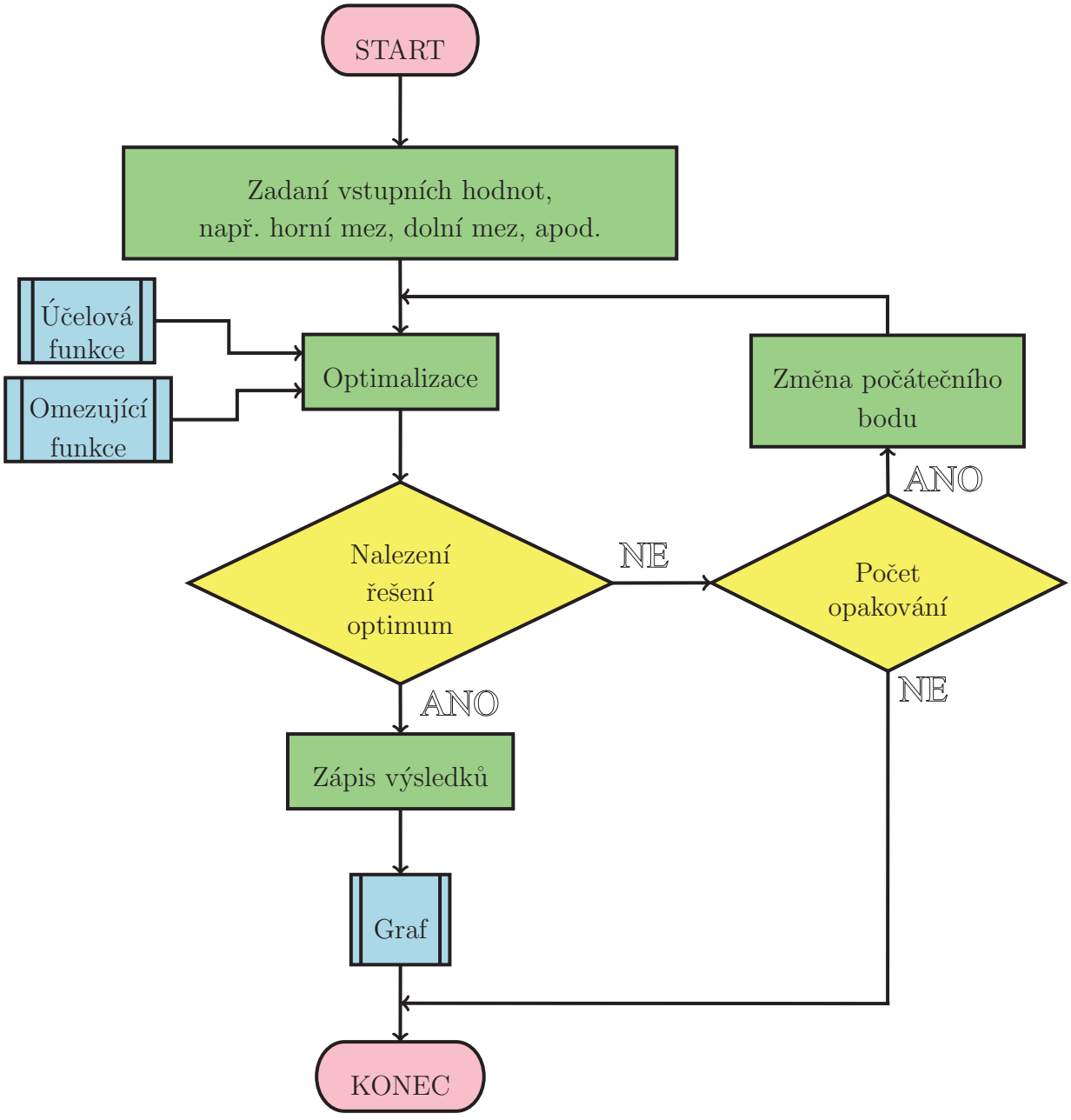
5. Struktura výpočtu

   Na začátku výpočtu nastavíme výše uvedené body (1 až 4) a spustíme samotnou optimalizaci. Pomocí proměnné *hodnota* zjistíme, zdali bylo nalezeno přípustné řešení, anebo se algoritmus zastavil z důvodu výpočetních omezení (maximální počet výpočtů). Jestli se *hodnota* $= 1$, je optimum nalezeno a přejde se k vykreslování diagramu. Ale jestli nebylo optimum nalezeno, provede se změna nastavení počátečního bodu a optimalizace se spustí znovu. Logicky je zřejmé, že tato smyčka by se mohla provádět vícekrát, a proto je nadefinovaná podmínka pro počet opakování. Tento počet je předem nastaven na maximální hodnotu rovnající se 10. Když bude překročena tato hodnota, výpočet se zastaví a nezobrazí se diagram. Struktura výpočtu je celá znázorněna v obrázku 3.1.

   Výsledek nemusí být optimální řešení, i když bude splněna podmínka pro optimum. Tento jev nastane, když se nalezne optimální řešení (výpočetně), ale toto řešení

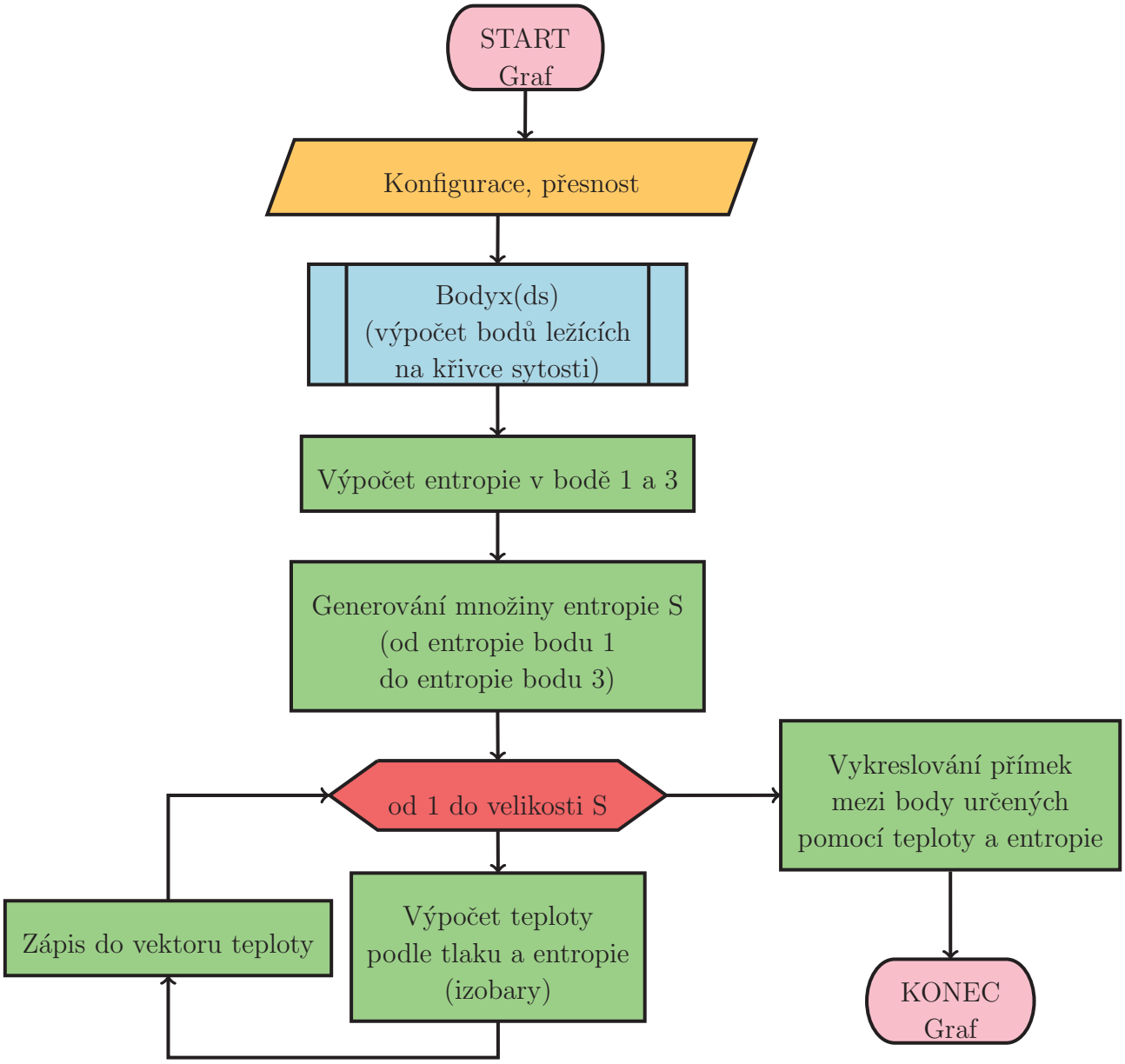
je reálně nesmyslné. Zmíněný nedostatek se ale projeví v t-s diagramu, a tak se výsledky dají jednoduše oddělit.

Výpočet probíhá následovně:



Obrázek 3.1: Vývojový diagram výpočtu ideálního R-C cyklu

Struktura výpočtu účelové funkce a omezující funkce je zřejmá z předchozího vysvětlení, a proto se budeme dále zabývat jen strukturou vykreslování t-s diagramu (obrázek 3.2).



Obrázek 3.2: Vývojový diagram vykreslování t-s diagramu

Výsledná konfigurace pak vypadá následovně:

	n_1 $[bar]$	n_2 $[°C]$	n_3 $[bar]$
Počáteční hodnoty	170	565	0, 08
Výsledné hodnoty	120, 27	565	0, 8
Výsledná termická účinnost			42, 35%

Tabulka 3.1: Tabulka výsledku optimalizace ideálního cyklu, varianta 1

Jelikož výsledek řešení je lokálním extrémem, zavedeme další počáteční bod (tabulka 3.2) a vypočteme. Tímto postupem se pokusíme určit přibližnou pozici globálního extrému.

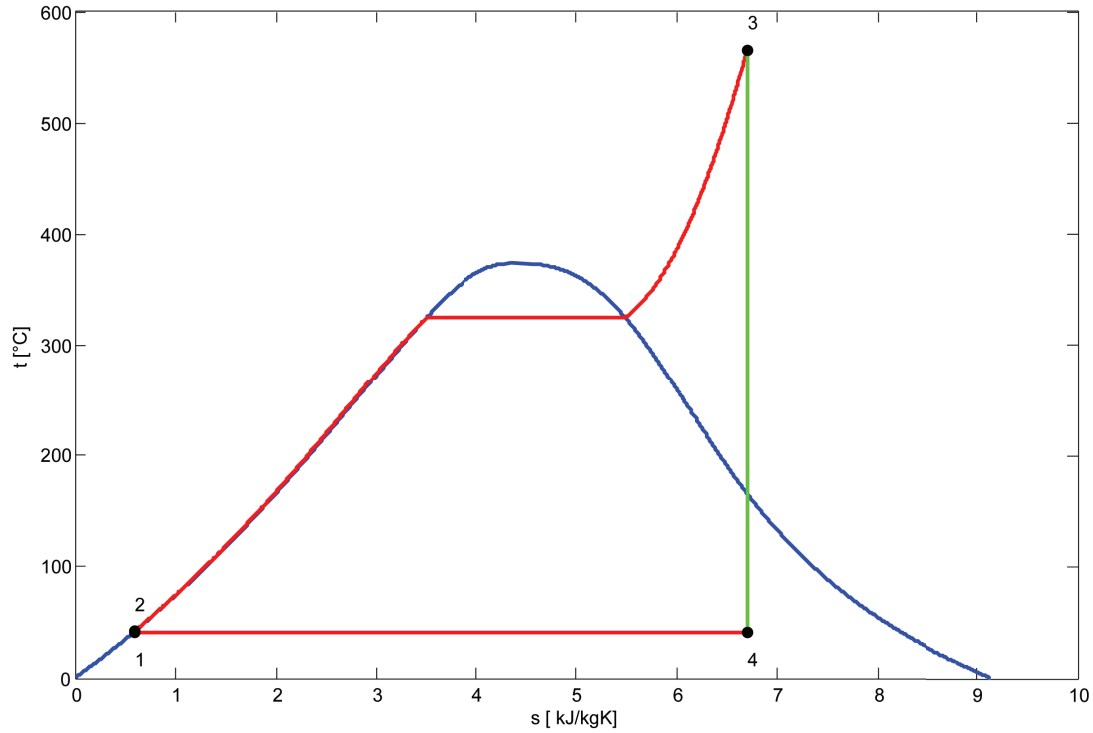
	počáteční body					
	I	II	III	IV	V	VI
n_1 $[bar]$	150	120	100	70	50	30
n_2 $[°C]$	500	400	350	300	250	200
n_3 $[bar]$	1	2	4	7	10	15

Tabulka 3.2: Počáteční hodnoty

Počáteční	Výsledné hodnoty			
hodnoty	n_1 $[bar]$	n_2 $[°C]$	n_3 $[bar]$	Termická účinnost[%]
I	120, 27	565	0, 08	42, 35
II	120, 27	565	0, 08	42, 35
III	120, 27	565	0, 08	42, 35
IV	120, 27	565	0, 08	42, 35
V	120, 27	565	0, 08	42, 35
VI	120, 27	565	0, 08	42, 35

Tabulka 3.3: Tabulka výsledku optimalizace ideálního cyklu, varianta 2

Jak je vidět z tabulky 3.3, při jiném nastavení počátečního bodu nám vychází stejné výsledky. To znamená, že jsme se přiblížili ke globálnímu extrému či ho přímo našli pro naše dané podmínky. Ideální konfigurace je zapsána v tabulce 3.1 a její t-s diagram je následující:



Obrázek 3.3: t-s diagram ideálního R-C cyklu

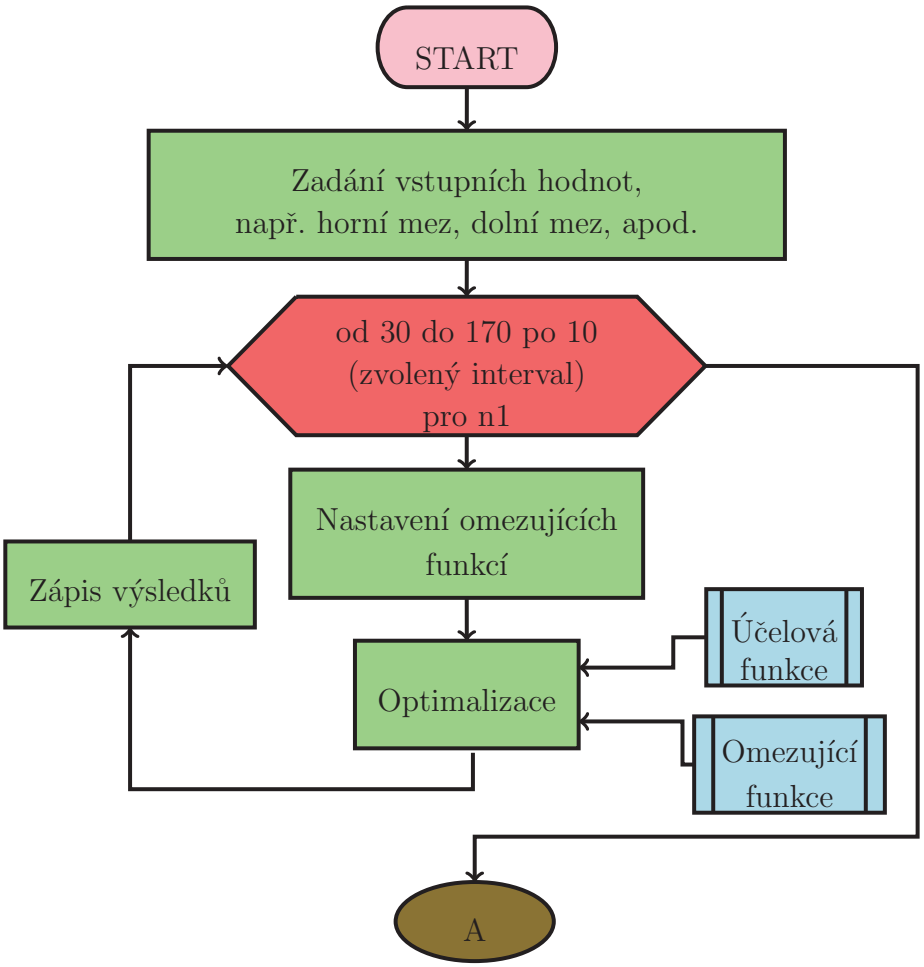
3.2. Citlivostní analýza ideálního cyklu

Jelikož příklad 3.1 se zabýval pouze nalezením ideální konfigurace, provedeme citlivostí analýzu. Neboli budeme posuzovat, jakou má tendenci výsledná hodnota účelové funkce modelu 3.6, jestliže nastavíme jednu z proměnných na přesné číslo. Toto číslo budeme postupně zvětšovat a zjišťovat, zdali se nám výsledná hodnota účelové funkce zvětšuje, či zmenšuje. Hodnoty budeme nastavovat z intervalu 3.7 , který si zvolíme podle předchozích výsledků.

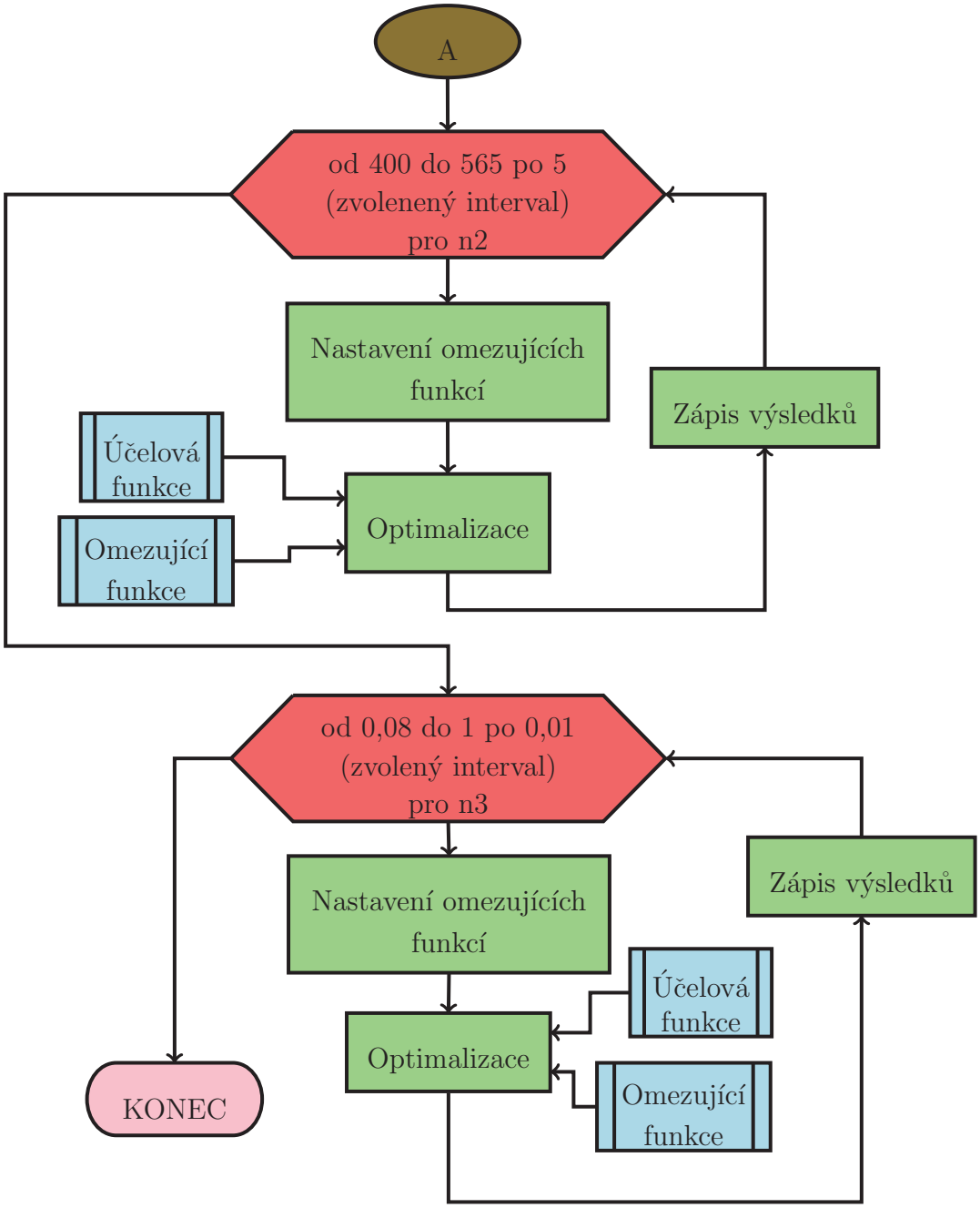
$$\begin{aligned} n_1 &\in < 30; 170 > \\ n_2 &\in < 300; 560 > \\ n_3 &\in < 0,08; 1 > \end{aligned} \tag{3.7}$$

Tuto modifikaci lze realizovat dvěma následujícími způsoby. Jedním ze způsobů je snížení počtu proměnných a s tím související přepsání celého modelu. Tato varianta není příliš vhodná z důvodu nutnosti přepsání modelu pro všechny proměnné. Proto zvolíme druhou variantu modifikace, jež je přidání omezení, které nám jednu proměnnou zafixuje na předem danou hodnotu např. $n_1 = 100$. U této varianty se mezi citlivostí pro jednotlivé proměnné mění jen podmínka, kterou budeme fixovat danou proměnnou.

Úprava kódu se projeví zavedením cyklu *for* a přidáním omezující funkce ve tvaru rovnice (viz příloha C). Výpočet má pak následující strukturu:



Obrázek 3.4: Vývojový diagram výpočtu citlivosti ideálního R-C cyklu



Obrázek 3.4: Vývojový diagram výpočtu citlivosti ideálního R-C cyklu - pokračování

3.2.1. Závislost výsledné hodnoty účelové funkce na proměnné n_1

Abychom zjistili zmíněnou závislost, budeme hledat nejprve citlivost proměnné n_1, která nám udává hodnotu tlaku při ohřevu. Výsledky citlivostní analýzy jsou znázorněny v tabulce 3.4, kde jsou zapsány již výsledné optimální konfigurace.

Z tabulky vyplývá, že z počátku se se zvyšujícím tlakem zvyšuje i účinnost. Jenže tahle tendence se obrací na hodnotě 120 *bar*. Tento bod obratu nastává z důvodu nutnosti zvýšení tlaku kondenzace, které je zapříčiněno nutností splnit omezující podmínky pro minimální suchost páry na výstupu z turbíny.

n_1 $[bar]$	n_2 $[°C]$	n_3 $[bar]$	výsledná termická účinnost [%]
30	565	0,08	37,33
40	565	0,08	38,44
50	565	0,08	39,29
60	565	0,08	39,97
70	565	0,08	40,52
80	565	0,08	41,00
90	565	0,08	41,40
100	565	0,08	41,76
110	565	0,08	42,07
120	565	0,08	42,35
130	565	0,10	42,12
140	565	0,12	41,87
150	565	0,14	41,62
160	565	0,17	41,37
170	565	0,20	41,11

Tabulka 3.4: Citlivostní analýza proměnné n_1

3.2.2. Závislost výsledné hodnoty účelové funkce na proměnné n_2

Druhou proměnnou modelu 3.6 je n_2, tato proměnná nám udává maximální teplotu přehřáté páry. Maximální teplota je určena několika kritérii. Nejdůležitější kritérium je cena, od které se odvíjí použitý materiál a technologie. Výsledné konfigurace jsou znázorněny v tabulce 3.5.

n_1 $[bar]$	n_2 $[°C]$	n_3 $[bar]$	výsledná termická účinnost [%]
45,53	400	0,08	36,49
47,03	405	0,08	36,69
48,56	410	0,08	36,88
50,13	415	0,08	37,07
51,75	415	0,08	37,26
⋮	⋮	⋮	⋮
107,90	545	0,08	41,70
110,89	550	0,08	41,87
113,95	555	0,08	42,03
117,07	560	0,08	42,19
120,27	565	0,08	42,35

Tabulka 3.5: Citlivostní analýza proměnné n_2

Tabulka 3.5 ukazuje závislost mezi maximální teplotou přehřáté páry, tlakem ohřívání a termickou účinností. Čím větší je maximální teplota přehřáté páry, tím větší je i tlak ohřívání (ideální tlak viz. kapitola 3.2.1). To vyplývá z podmínky pro minimální hodnotu

suchosti páry na výstupu z turbíny. Podobná závislost platí i pro maximální teplotu přehřáté páry a termickou účinnost (čím větší teplota, tím větší účinnost).

3.2.3. Závislost výsledné hodnoty účelové funkce na proměnné n_3

Poslední proměnou je n_3, tedy tlak kondenzace. Výsledky citlivostní analýzy jsou zapsané v tabulce 3.6.

n_1 $[bar]$	n_2 $[°C]$	n_3 $[bar]$	výsledná termická účinnost [%]
$120,27$	565	$0,08$	$42,35$
$126,14$	565	$0,09$	$42,21$
$131,56$	565	$0,1$	$42,08$
$136,59$	565	$0,11$	$41,96$
$141,28$	565	$0,12$	$41,84$
⋮	⋮	⋮	⋮
170	565	$0,96$	$36,38$
170	565	$0,97$	$36,34$
170	565	$0,98$	$36,31$
170	565	$0,99$	$36,27$
170	565	1	$36,24$

Tabulka 3.6: Citlivostní analýza proměnné n_3

Výsledkem je, že čím větší je hodnota tlaku při kondenzaci, tím menší je účinnost. Tuto tendenci nezmění ani zvyšující se tlak ohřívání.

3.3. Nadkritický ideální cyklus

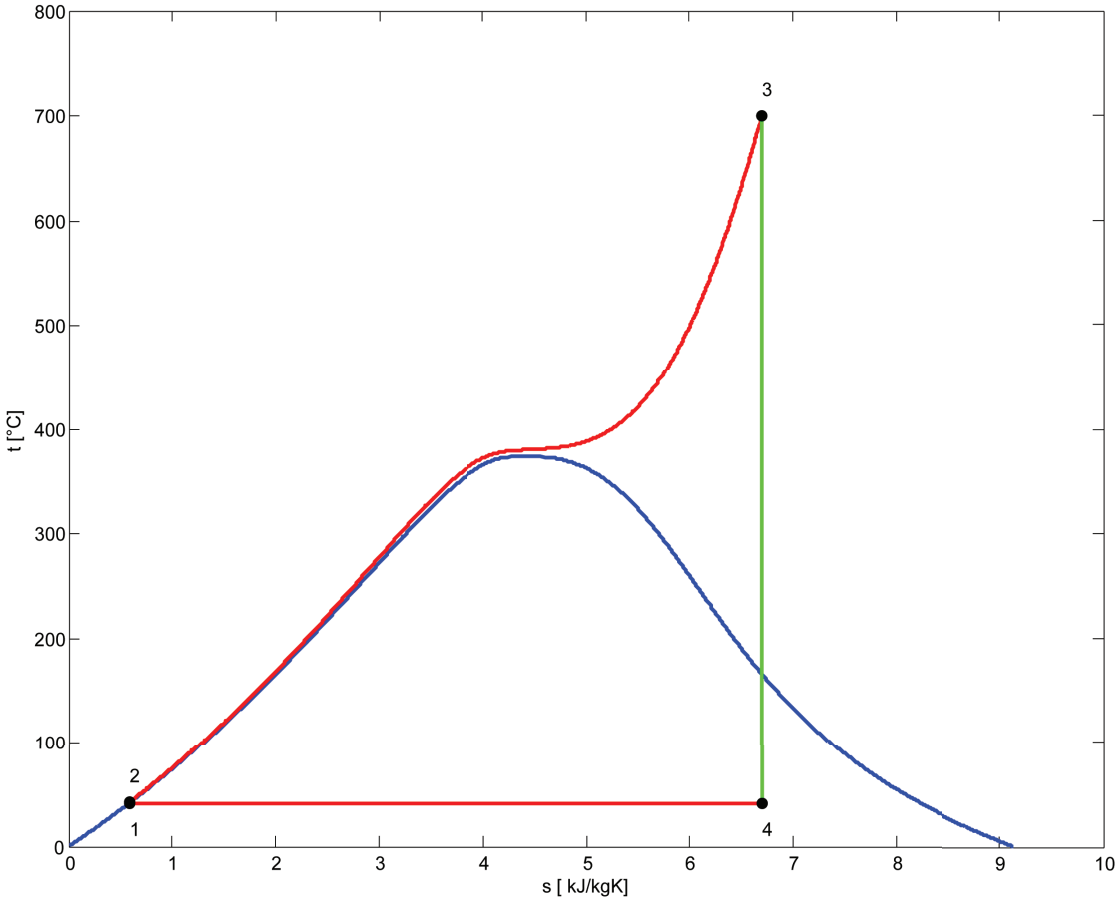
U nadkritického cyklu povolíme překročení kritického bodu. Z modelu 3.6 vypustíme horní mez pro proměnnou n_1 a pro n_2 horní mez nastavíme na hodnotu 700 $°C$. Program v příloze B se změní na hodnotě horního i dolního vektoru a na počátečním bodě iterace. Výsledek nadkritického oběhu je zapsán v tabulce 3.7.

	n_1 $[bar]$	n_2 $[°C]$	n_3 $[bar]$
Počáteční body	300	500	1
Výsledné hodnoty	$237,63$	700	$0,8$
Výsledná termická účinnost			$46,41\%$

Tabulka 3.7: Tabulka výsledku optimalizace nadkritického cyklu

Výsledkem je lokální extrém, ale jak bylo dokázáno v příkladu 3.1, pro nalezení globálního extrému nemusí být zaváděny další počáteční body. Výsledný t-s diagram je znázorněn na obrázku 3.5.

Jak je vidět z porovnání výsledku 3.7 a 3.1, účinnost nám stoupla o více jak 4%. Tato skutečnost je příznivá, ale dovolení překročení nakritického bodu zároveň s sebou nese velké problémy na konstrukčních řešení a materiálech.



Obrázek 3.5: t-s diagram nadkritického oběhu

3.4. Cyklus s jedním příhřevem

Jednou ze základních úprav R-C cyklu je zavedení příhřevu (jeho realizace viz kapitola 2). Z diagramu 2.5 vyplývá, že:

- dodaná energie se rovná

$$q_{in} = i_3 - i_2 + i_5 - i_4, \tag{3.8}$$

- získaná energie na turbínách

$$a_T = i_3 - i_4 + i_5 - i_6, \tag{3.9}$$

- odvedená energie v čerpadle

$$a_{\check{C}} = i_2 - i_1, \tag{3.10}$$

- termická účinnost

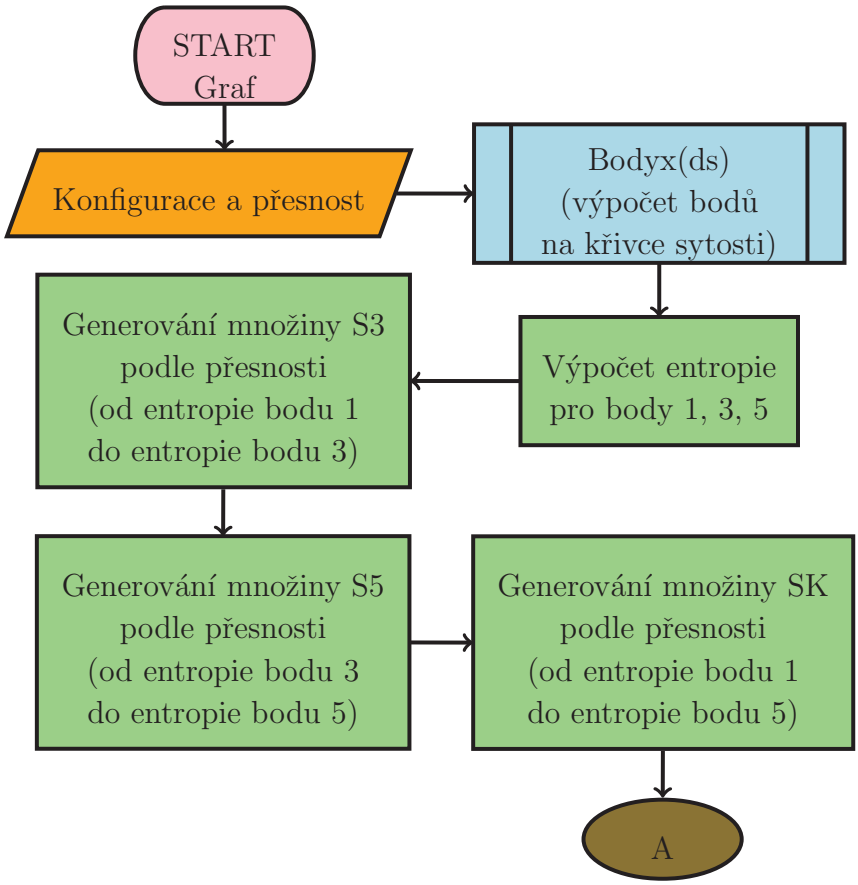
$$\eta_t = \frac{a_t - a_{\check{C}}}{q_{in}} = \frac{i_3 - i_4 + i_5 - i_6 - (i_2 - i_1)}{i_3 - i_2 + i_5 - i_4}. \tag{3.11}$$

Proměnné označíme podobně jako u příkladu 3.1, a to tedy tak, že n_1 je tlak při prvním ohřívání, n_2 je teplota v bodě 3, n_3 je tlak při příhřevu, n_4 je teplota v bodě 5 a n_5 je tlak kondenzace. Omezující funkce jsou taktéž podobné, pouze se přidá podmínka pro minimální hodnotu suchosti páry na výstupu z první části turbíny a dále se přidají velikostní omezení pro proměnné n_3 a n_4. Sestavený model poté vypadá následovně:

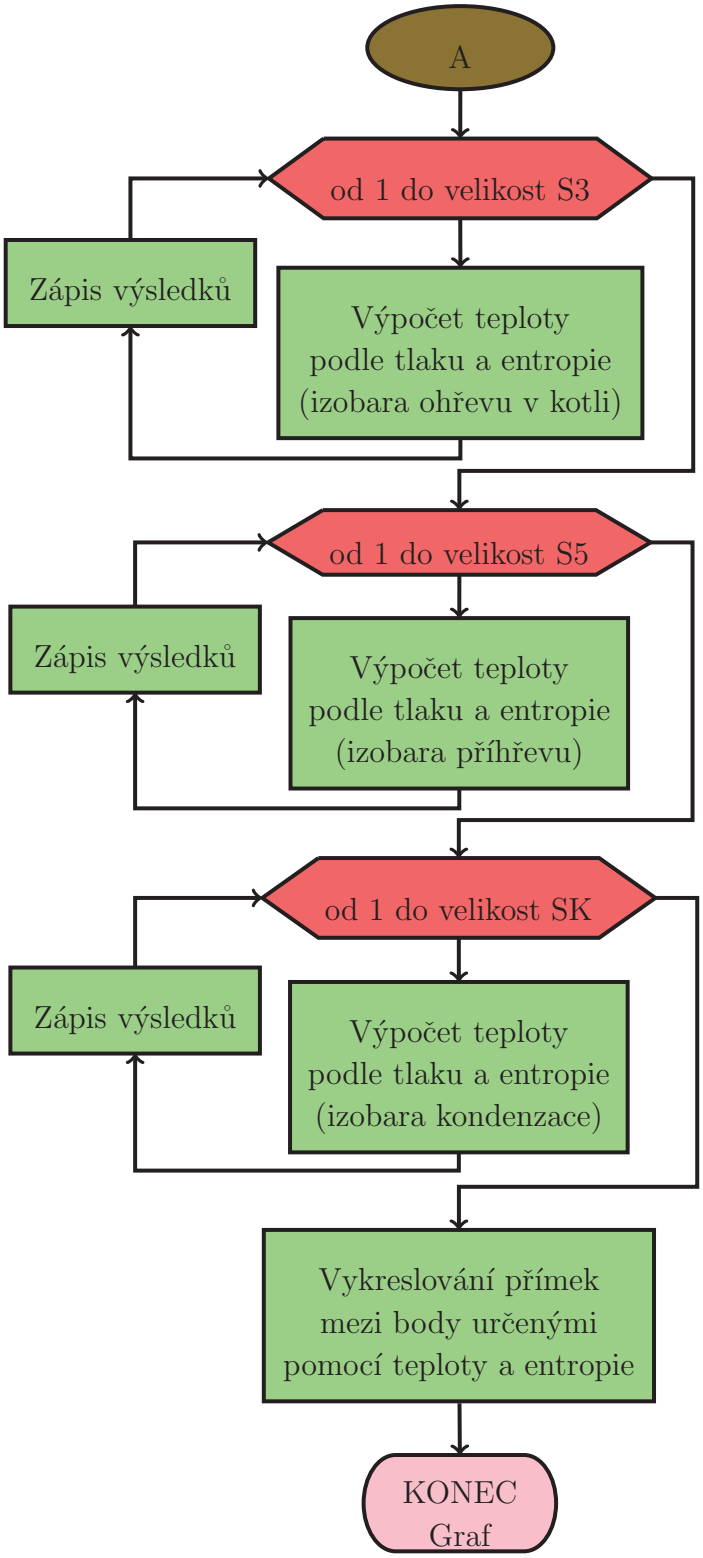
$$\textbf{min} \quad \frac{i_3(n_1;n_2)-i_4(n_3;s_3(n_1;n_2))+i_5(n_3;n_4)-i_6(n_5,s_5(n_3,n_4))-i_2(n_1;s_1(n_5^{'}))+i_1(n_5^{'})}{i_2(n_1;s_1(n_5^{'}))-i_3(n_1;n_2)-i_5(n_3;n_4)+i_4(n_3;s_3(n_1;n_2))} \tag{3.12}$$

$$\begin{aligned} \textit{za podmínek} \quad x_6(n_5; s_5(n_3; n_4)) &\geq 0,8 \\ x_4(n_3; s_3(n_1; n_2)) &\geq 0,8 \\ x_1(n_5; s_1(n_5^{'}) &= 1 \\ n_1 &\leq 170 \\ n_2 &\leq 565 \\ n_3 &\leq 100 \\ n_4 &\leq 565 \\ n_5 &\geq 0,08 \\ n_1,\ n_2,\ n_3,\ n_4,\ n_5 &\geq 0. \end{aligned}$$

Výpočet probíhá podobným způsobem jako u předchozích úloh. Jediná změna je v kreslení diagramu. Ten probíhá následující strukturou:



Obrázek 3.6: Vývojový diagram vykreslování t-s diagramu R-C cyklu s jedním příhřevem



Obrázek 3.6: Vývojový diagram vykreslování t-s diagramu R-C cyklu s jedním příhřevem - pokračování

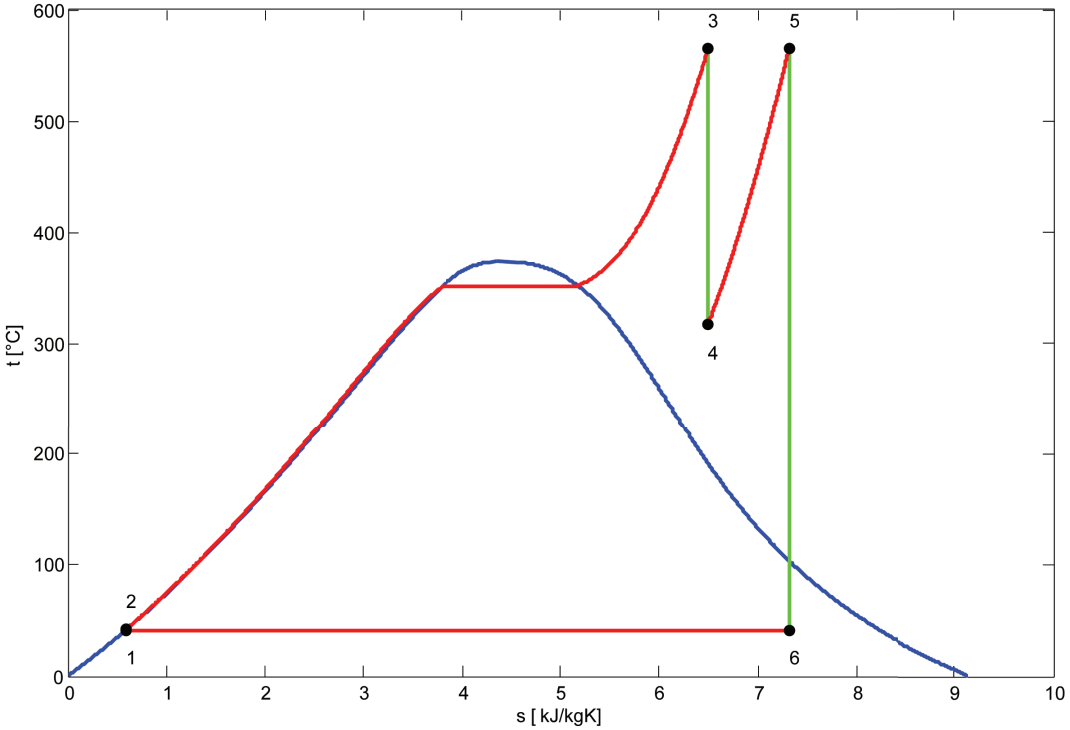
Výsledná konfigurace je lokálním extrémem řešení. Aby byl nalezen globální extrém, použijeme stejný postup jako v předchozích příkladech. A to tak, že po zadání více

počátečních bodů je globální extrém zapsán v tabulce 3.8 a jeho t-s diagram znázorněn na obrázku 3.7.

	$n_1\ [bar]$	$n_2\ [°C]$	$n_3\ [bar]$	$n_4\ [°C]$	$n_5\ [bar]$
Počáteční body	100	565	50	565	1
Výsledné hodnoty	170	565	37,01	565	0,08
Výsledná termická účinnost			45,17%		

Tabulka 3.8: Tabulka výsledku optimalizace R-C cyklu s jedním přihříváním

Z výsledné konfigurace vychází, že tlak mezi body 2 a 3 je maximální dovolený. A to z důvodu, že díky přihřívání nám bod 6 vychází nad křivku syté páry, a tím zaručí podmínku pro suchost páry. To vše se děje, i když jsme na maximálním tlaku ohřívání a minimálním tlaku kondenzace. Díky této vlastnosti je výsledná termická účinnost vyšší bez přihřívání, a to o více jak 4%, ale zároveň nižší než u nadkritického oběhu.



Obrázek 3.7: t-s diagram R-C cyklu s jedním příhřevem

3.5. R-C cyklus s více příhřevy

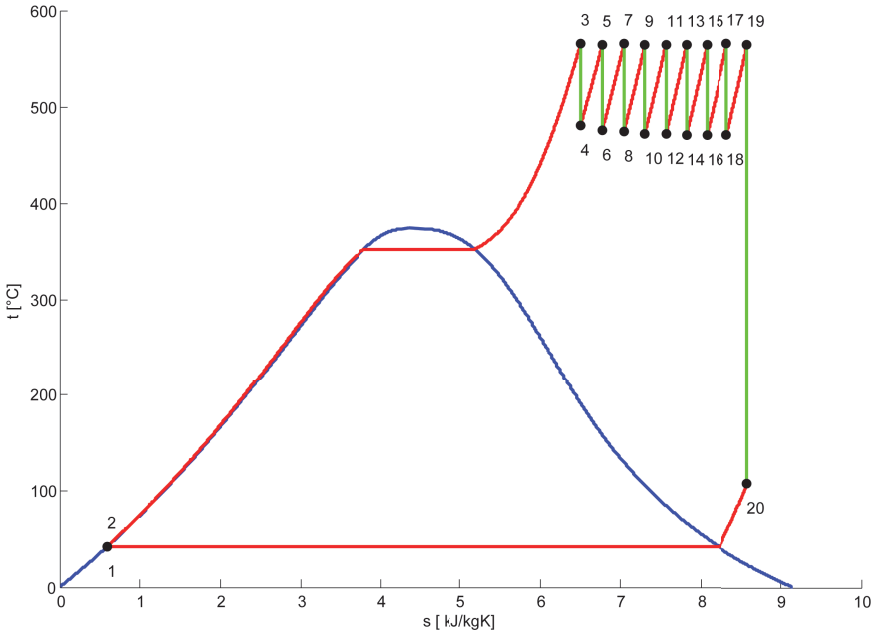
Výsledek příkladu 3.4 nám odůvodňuje realizaci přihřívání, ale neříká už nic blíže o optimálním počtu příhřevů a jejich optimální velikosti. Tato otázka je komplikovaná, a tak ji rozdělíme do dvou sekcí. První sekce se bude zabývat ideální velikostí jednotlivých přihřívání pro jeden předem daný počet. A druhá část popisuje, jaký je ideální počet přihřívání. Na závěr výsledky těchto dvou sekcí spojíme, a získáme tak požadované ideální řešení.

Hlavní změnou provedenou na modelu není ani tak zápis modelu (ten je intuitivní), ale přepis kódu (příloha E). Tento kód je zapsaný pro obecný počet přihřívání, a tudíž je

v porovnání s ostatními kódy méně přehledný. Z tohoto důvodu si ho popíšeme v jednotlivých krocích:

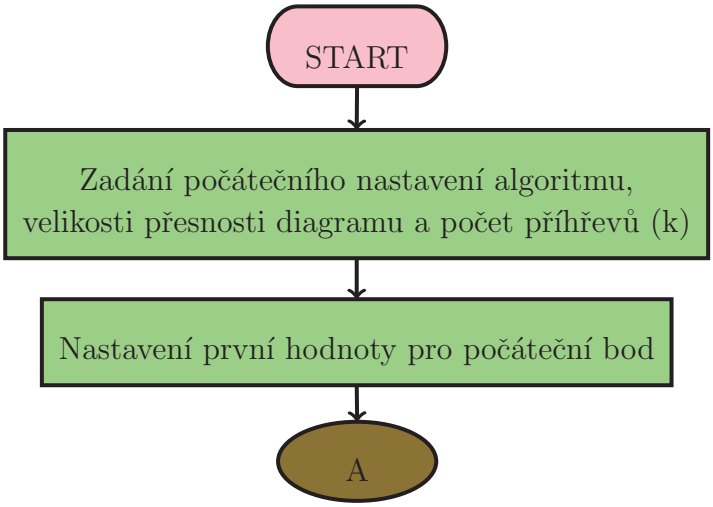
1. Nastavení počtů příhřevů
   Nastavení počtu příhřevů se provádí pomocí globální proměnné k.

2. Změna označení proměnných
   Proměnnou n_1 je označený tlak kondenzace, n_2 označení tlaku pro ohřívání v kotli, n_3 pro teplotu v bodě 3, n_4 pro tlak prvního příhřevu, n_5 pro teplotu v bodě 5 atd. (viz obrázek 3.8).

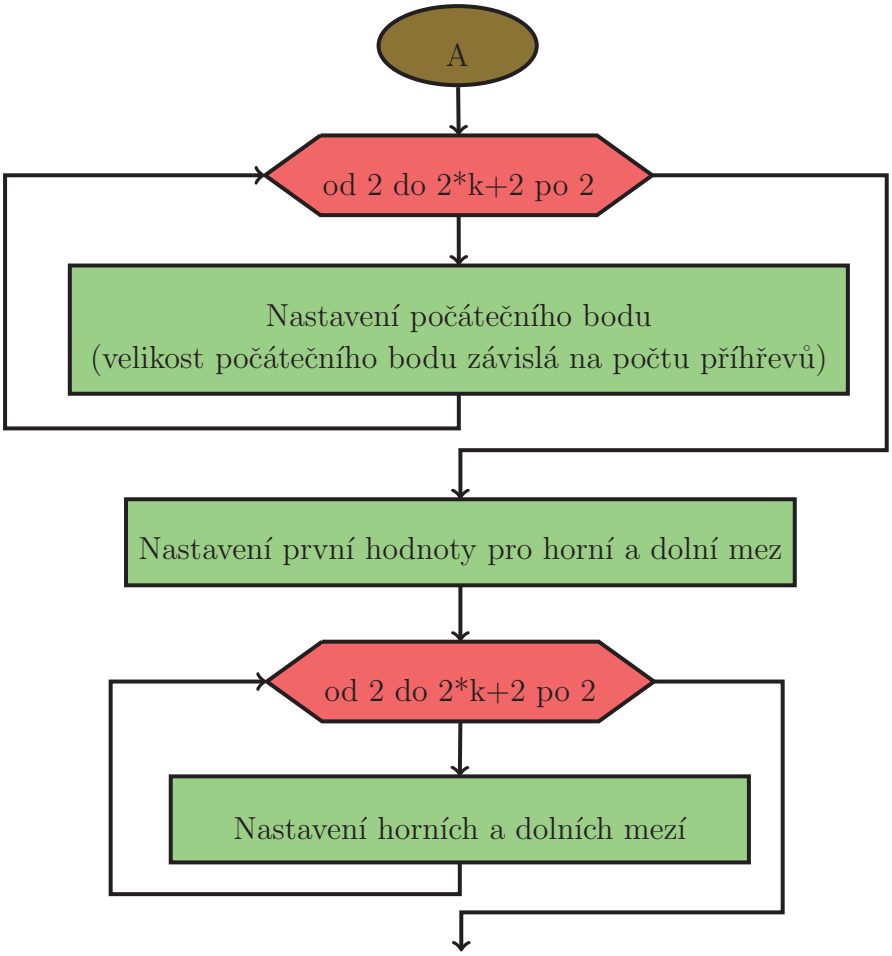


Obrázek 3.8: t-s diagram R-C cyklu se čtrnácti příhřevy

3. Struktura výpočtů
   Struktura výpočtů je shodná s obrázkem 3.1. Liší se pouze v *zadaní vstupních hodnot*. Obrázek 3.9 je znázornění této změny.

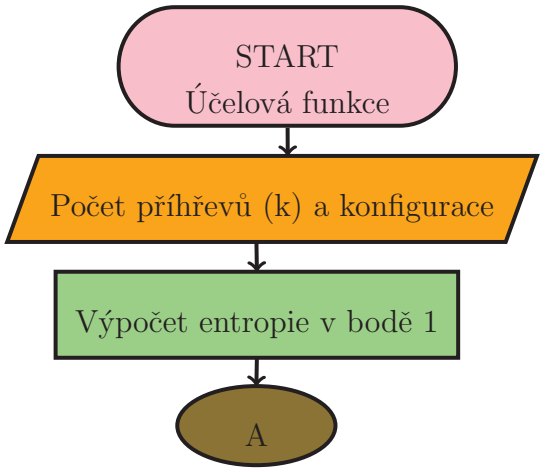


Obrázek 3.9: Vývojový diagram výpočtu R-C cyklu s více příhřevy

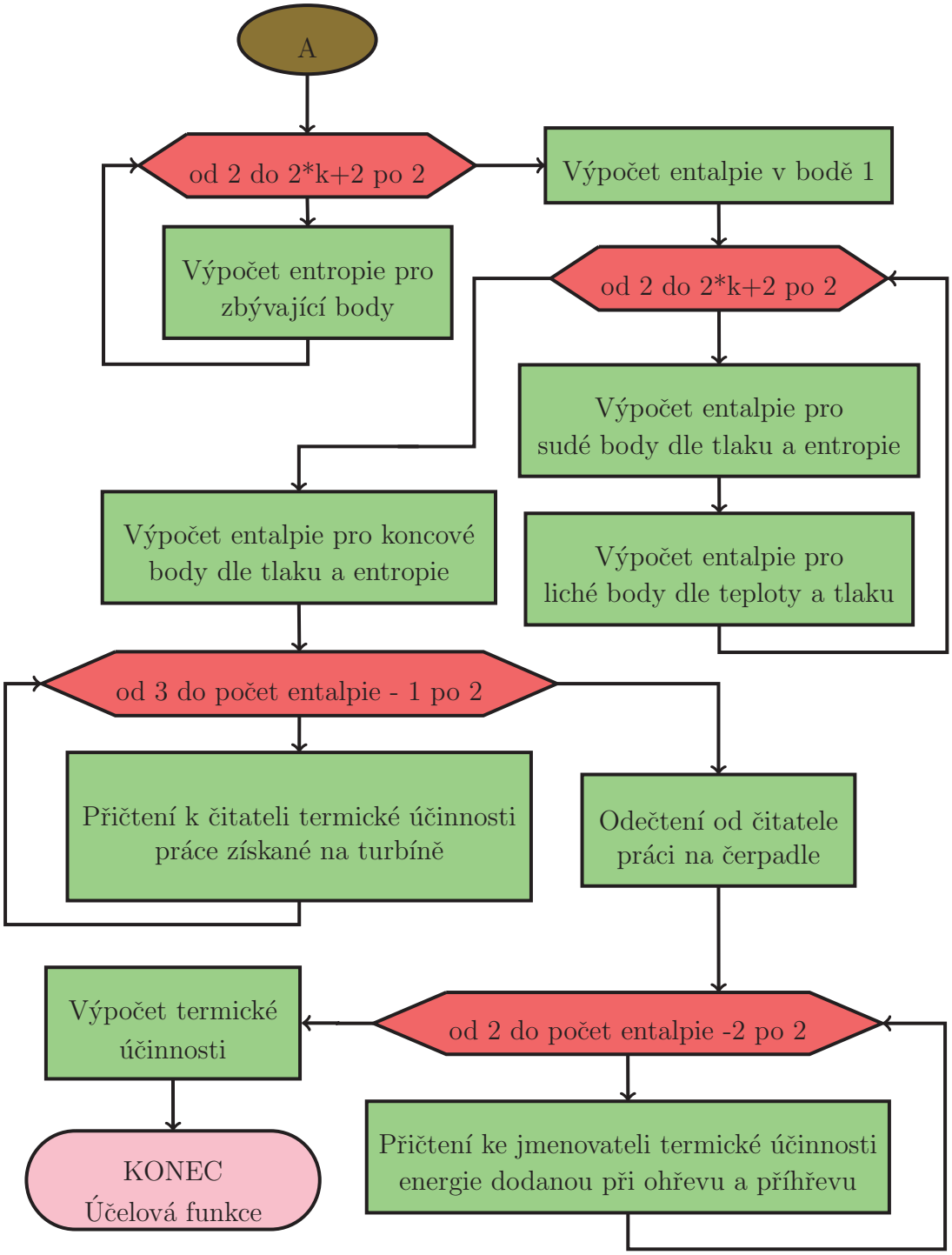


Obrázek 3.9: Vývojový diagram výpočtu R-C cyklu s více příhřevy - pokračování

4. Struktura účelové funkce a omezující funkce
   Další změnou je změna účelové funkce. Ta se mění z jednoduchého přepsání matematického modelu na strukturu znázorněnou v obrázku 3.10. Tato změna se projeví i u omezující funkce, ale jelikož její podstata je stejná jako u účelové funkce, znázorníme jen její změnu.

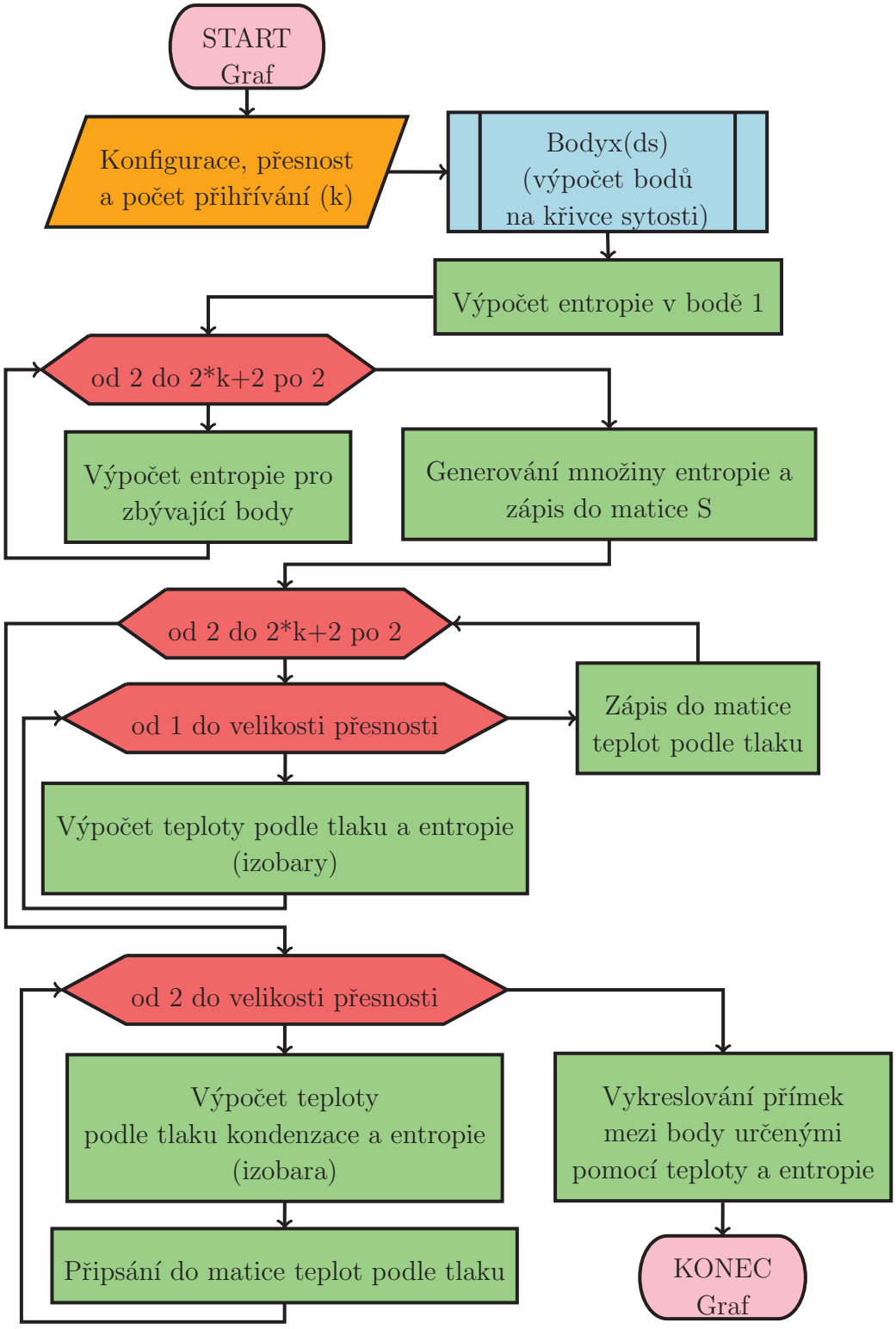


Obrázek 3.10: Vývojový diagram účelové funkce



Obrázek 3.10: Vývojový diagram účelové funkce R-C cyklu - pokračování

5. Struktura funkce *graf*
   Struktura vykreslování t-s diagramu se také změní. V obrázku 3.6 (znázornění grafu pro jeden příhřev) jsou zapsány množiny entropie každá samostatně (každá má své pojmenování), ale v tomto případě to není možné realizovat, jelikož nevíme počet přihřívání. Z tohoto důvodu jsou všechny množiny entropie zapsány do jedné matice. Každý řádek této matice nám udává jednu množinu entropie. Tato změna platí i pro vypočtenou teplotu, a proto obrázek 3.11 znázorňuje tuto změnu jen u teploty.



Obrázek 3.11: Vývojový diagram grafu R-C cyklu s více příhřevy - pokračování

3.5.1. Optimalizace velikosti příhřevů

Optimální velikost příhřevu budeme určovat pro konečnou hodnotu příhřevu. Tato hodnota je nastavena na 5. To proto, aby byl počet stupňů poměrně velký, ale zároveň,

aby nám počet proměnných příliš nenarostl. Když už máme nastavený počet příhřevů, hledáme globální extrém úlohy. K nalezení globálního extrému použijeme počáteční body z tabulky 3.9. Výsledné konfigurace pro jednotlivé počáteční body zapíšeme do tabulky 3.10.

	počáteční body				
	I	II	III	IV	V
n_1 $[bar]$	0,08	0,08	0,08	0,08	0,08
n_2 $[bar]$	150	200	140	118	110
n_3 $[°C]$	565	565	565	400	400
n_4 $[bar]$	150	180	130	116	100
n_5 $[°C]$	565	565	565	400	400
n_6 $[bar]$	150	160	120	114	90
n_7 $[°C]$	565	565	565	400	400
n_8 $[bar]$	150	140	110	112	80
n_9 $[°C]$	565	565	565	400	400
n_{10} $[bar]$	150	120	100	110	70
n_{11} $[°C]$	565	565	565	400	400
n_{12} $[bar]$	150	100	90	108	60
n_{13} $[°C]$	565	565	565	400	400

Tabulka 3.9: Tabulka počátečních bodů

Výsledné hodnoty	Počáteční body				
	I	II	III	IV	V
n_1 $[bar]$	0,08	0,08	0,08	0,08	0,08
n_2 $[bar]$	170	170	170	170	170
n_3 $[°C]$	565	565	565	565	565
n_4 $[bar]$	82,32	82,32	82,32	82,32	82,32
n_5 $[°C]$	565	565	565	565	565
n_6 $[bar]$	37,47	37,47	37,47	37,47	37,47
n_7 $[°C]$	565	565	565	565	565
n_8 $[bar]$	16,5	16,5	16,5	16,5	16,5
n_9 $[°C]$	565	565	565	565	565
n_{10} $[bar]$	7,15	7,15	7,15	7,15	7,15
n_{11} $[°C]$	565	565	565	565	565
n_{12} $[bar]$	3,08	3,08	3,08	3,08	3,08
n_{13} $[°C]$	565	565	565	565	565
Termická účinnost[%]	48,23	48,23	48,23	48,23	48,23

Tabulka 3.10: Tabulka výsledku optimalizace velikosti příhřevů

Protože všechny výsledné konfigurace vychází stejné, je velmi pravděpodobné, že jsme nalezli globální extrém. Díky této skutečnosti můžeme posuzovat ideální velikost příhřevu na ideálním řešení. Posuzování provedeme pomocí t-s diagramu a *X Steamu*. *X Steam* použijeme na vypočtení jednotlivých prací a z t-s diagramu (obrázek 3.13) získáme určitý nadhled do problému a to tak, že vizuálně porovnáme změny velikostí.

Velikost jednotlivých prací na turbíně jsou následující:

- mezi body 3 a 4

$$a_{3,4} = i_3 - i_4 = 233,65\ kJ/kg, \tag{3.13}$$

- mezi body 5 a 6

$$a_{5,6} = i_5 - i_6 = 266,95\ kJ/kg, \tag{3.14}$$

- mezi body 7 a 8

$$a_{7,8} = i_7 - i_8 = 254,73\ kJ/kg, \tag{3.15}$$

- mezi body 9 a 10

$$a_{9,10} = i_9 - i_{11} = 293,12\ kJ/kg, \tag{3.16}$$

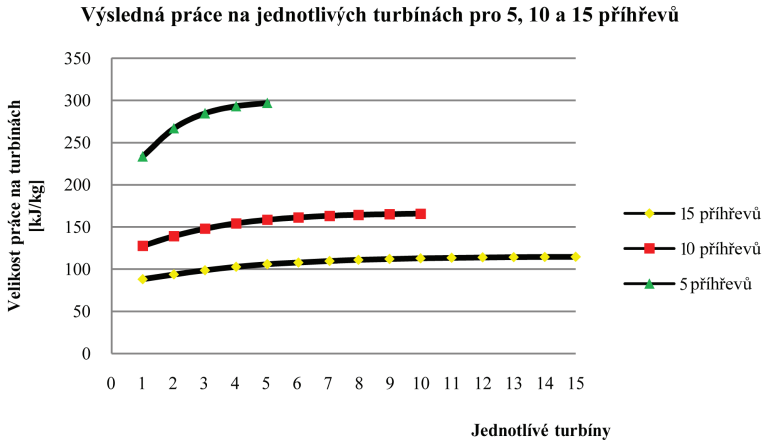
- mezi body 11 a 12

$$a_{11,12} = i_{11} - i_{12} = 296,92\ kJ/kg, \tag{3.17}$$

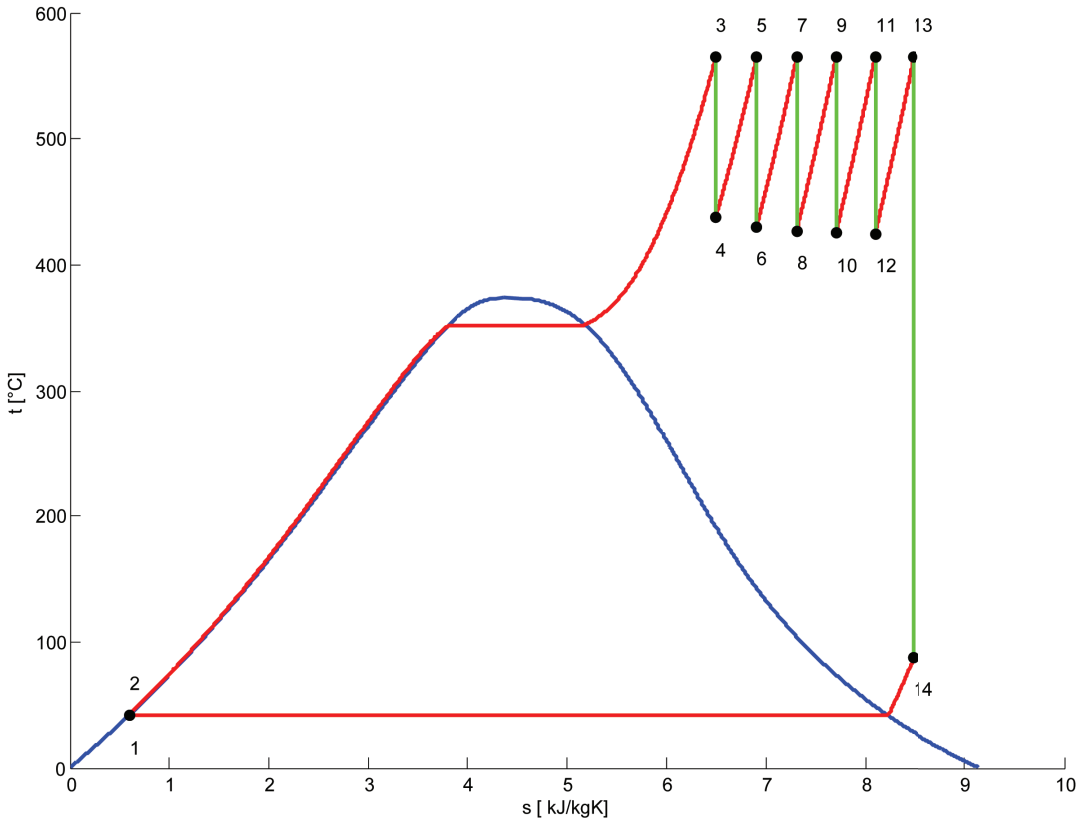
- mezi body 13 a 14

$$a_{13,14} = i_{13} - i_{14} = 962,53\ kJ/kg. \tag{3.18}$$

Z těchto výsledků vychází, že velikost jednotlivých výkonů se postupně zvětšují, až poslední zůstávají skoro nezměněny.



Obrázek 3.12: Výsledná práce na jednotlivých turbínách pro 5, 10 a 15 příhřevů



Obrázek 3.13: t-s diagram optimalizace velikosti příhřevu

Z grafu 3.12 vyplývá, že čím více použijeme příhřevu, tím je velikost jednotlivých prací menší a zároveň se změna výkonu zmenšuje. Změna je ale vizuální klam, jelikož procentuální změna (rozdíl prací na jednotlivých turbínách ku práci na první turbíně) se příliš nemění, jak je vidět v tabulce 3.11.

a_T $[kJ/kg]$	procentuální změna	a_T $[kJ/kg]$	procentuální změna	a_T $[kJ/kg]$	procentuální změna
233, 65		117, 02		88, 11	
266, 96	14, 3	126, 78	8, 3	93, 74	6, 4
284, 73	7, 6	134, 64	6, 7	98, 69	5, 6
293, 12	3, 6	140, 29	4, 8	102, 76	4, 6
296, 93	1, 6	144, 39	3, 5	105, 83	3, 5
		147, 15	2, 4	107, 81	2, 2
		149, 05	1, 6	109, 64	2, 1
		150, 36	1, 1	111, 00	1, 5
		151, 26	0, 8	112, 02	1, 2
		151, 86	0, 5	112, 81	0, 9
		152, 24	0, 3	113, 42	0, 7
				113, 88	0, 5
				114, 21	0, 4
				114, 44	0, 3
				114, 61	0, 2

Tabulka 3.11: Tabulka procentuálních změn výkonů na turbínách

Z předchozích zjištěných výsledků tedy můžeme stanovit závěr pro ideální velikost, jež nám udávají jednotlivé práce na turbínách. Zmíněná ideální velikost je proměnlivá. To znamená, že velikost postupně narůstá, až se ustálí.

Jenže praktická literatura se zmiňuje o konstantní velikosti, a tak zavedeme do optimalizační úlohy tuto podmínku a budeme sledovat změnu výsledné termické účinnosti.

Matematicky zápis podmínky konstantní velikosti příhřevu je následovný:

$$a_{3,4} = a_{5,6} = \ldots = a_{k\cdot 2+1;k\cdot 2+2} \quad , \textit{kde k je počet příhřevů.} \tag{3.19}$$

Jeho převedení do kódu se projeví přidáním podmínkových rovnic do funkce *omezfunkce*.

Velikost jednotlivých prací na turbíně jsou pak tyto:

- mezi příhřevy

$$a = 274,46\ kJ/kg, \tag{3.20}$$

- mezi body 13 a 14

$$a_{13,14} = i_{13} - i_{14} = 963,47\ kJ/kg. \tag{3.21}$$

Jak je vidět z tabulky 3.12, výsledky jsou prakticky stejné, tudíž jsme našli dvě optima. V praxi je daleko jednoduší realizovat stejné práce na turbínách, a proto je tato varianta využívána.

	Termická účinnost
Bez podmínky	48, 2348
S podmínkou	48, 2215

Tabulka 3.12: Tabulka rozdílu termické účinnosti s podmínkou a bez podmínky konstantní velikosti

3.5.2. Optimalizace počtu příhřevů

Teoreticky optimální počet příhřevů je buď nekonečný, anebo v řádech milionů a více. Ale tato vlastnost je technicky nerealizovatelná, a proto se zabýváme konečným počtem příhřevů.

Optimalizační úloha je pak celočíselné programování. Celočíselné programování není vhodné pro jeho přílišnou složitost. Z tohoto důvodu převedeme úlohu na jinou, která již nebude celočíselná. To provedeme následujícím způsobem: místo optimalizování počtu příhřevu budeme optimalizovat jednotlivé příhřevy (přesněji řečeno od jednoho příhřevu do 25.). Za kritérium optima vezmeme největší bod, pro něj bude přírůstek na účinnosti ještě větší než hodnota ϵ. Tato hodnota je velmi závislá na více faktorech např. cena realizace, možnost realizace apod. Pro náš případ zvolíme $\epsilon = 0,5$, protože pro nižší hodnoty není adekvátní přínos na úkor náročnost realizace. Takto získáme optimální počet přihřívání.

Výsledky v tabulce jsou již zapsané pro nejlepší nalezený počáteční bod. To tedy znamená pro ideální velikost příhřevu.

počet přihříváků	výsledná termická účinnost	ϵ	počet přihříváků	výsledná termická účinnost	ϵ
1	$45,17\%$		13	$49,19\%$	$0,05$
2	$46,41\%$	$1,24$	14	$49,24\%$	$0,05$
3	$47,32\%$	$0,91$	15	$48,28\%$	$0,04$
4	**47,87%**	**0,55**	16	$49,31\%$	$0,04$
5	$48,22\%$	$0,36$	17	$49,35\%$	$0,03$
6	$48,47\%$	$0,25$	18	$49,37\%$	$0,03$
7	$48,65\%$	$0,18$	19	$49,40\%$	$0,03$
8	$48,80\%$	$0,14$	20	$49,42\%$	$0,02$
9	$48,91\%$	$0,11$	21	$49,44\%$	$0,02$
10	$49,00\%$	$0,09$	22	$49,46\%$	$0,02$
11	$49,07\%$	$0,08$	23	$49,48\%$	$0,02$
12	$49,14\%$	$0,06$	24	$49,50\%$	$0,01$
			25	$49,51\%$	$0,01$

Tabulka 3.13: Tabulka výsledné hodnoty termické účinnosti optimalizace R-C cyklu do 25 přihřívání

Z tabulky 3.13 vyplývá, že ideální počet přihřívání je 4. Také by se mohlo uvažovat o realizaci 3 příhřevů, a to tehdy, když by se zahrnula neideálnost práce na turbínách (nárůst entropie).

3.5.3. Ideální konfigurace

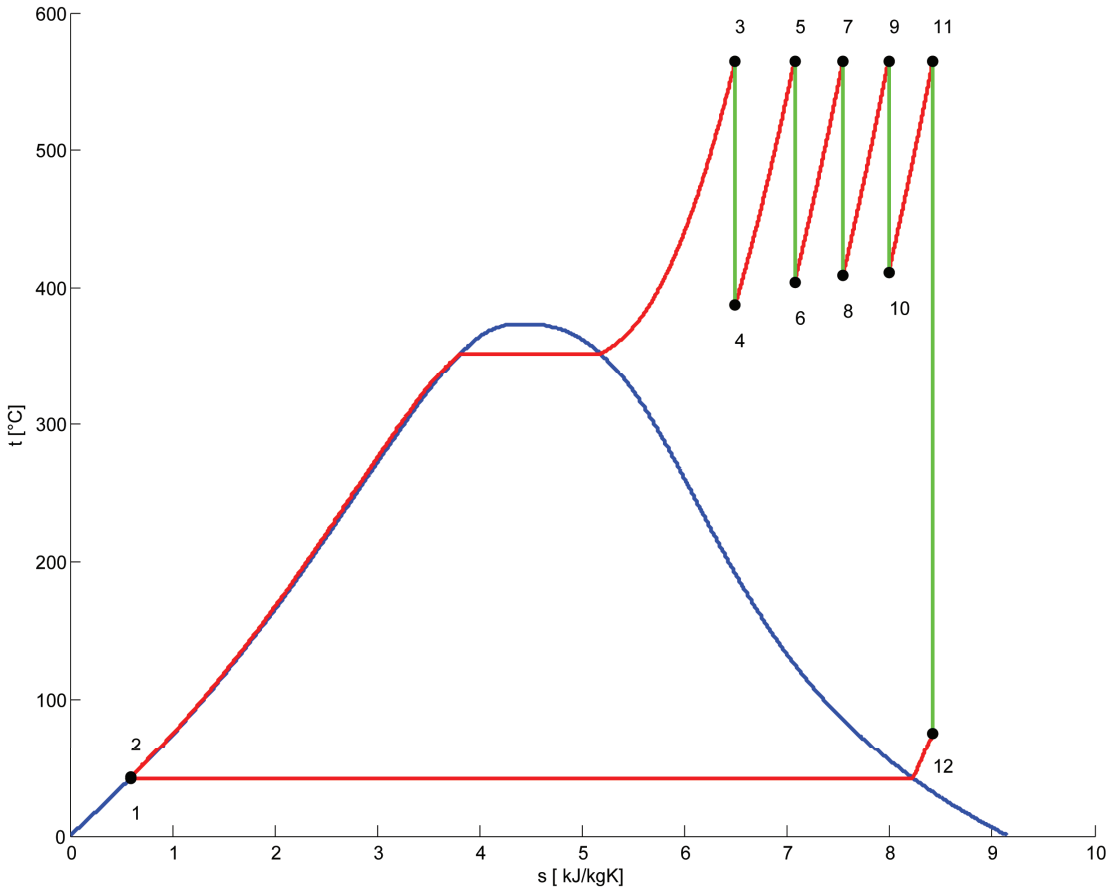
Hledání ideální konfigurace R-C cyklu s více příhřevy se rozdělilo do dvou optimalizačních úloh. První úloha řešila ideální velikost jednotlivých příhřevů, druhá optimální počet přihřátí. Jejich výsledky jsou následující:

- Ideální velikost jednotlivých příhřevů
  Ideální velikost je konstantní.
- Optimální počet příhřevů
  Ideální počet je 4.

Díky těmto závěrům můžeme stanovit nejlepší konfiguraci R-C cyklu s více příhřevy, která je zapsaná v tabulce 3.14. Výsledný t-s diagram je znázorněn v obrázku 3.14.

Výsledná konfigurace										
n_1 $[bar]$	n_2 $[bar]$	n_3 $[°C]$	n_4 $[bar]$	n_5 $[°C]$	n_6 $[bar]$	n_7 $[°C]$	n_8 $[bar]$	n_9 $[°C]$	n_{10} $[bar]$	n_{11} $[°C]$
$0,08$	170	565	$60,06$	565	$22,98$	565	9	565	$3,55$	565
Výsledná termická účinnost					$47,87$					

Tabulka 3.14: Tabulka výsledků optimalizace R-C cyklu s příhřevy



Obrázek 3.14: t-s diagram idálního R-C cyklu s přihříváním

Závěr

Diplomová práce byla zaměřena na optimalizaci Rankine-Clausiova cyklu, přesněji na hledání největší termické účinnosti při zavedení podmínek. Podmínky nahrazují omezení vyplývající z technických, cenových a materiálových vlastností. Pro hledání optimální konfigurace, bylo nutné sestavená simulačního modelu.

Diplomová práce se dělí na dvě části. První část byla věnována teoretickým informacím, které byly využity v druhé části. V oblasti optimalizace jsme se věnovali definicím základních pojmů, jimiž jsou lokální a globální extrém, lineární a nelineární programování, citlivostní analýza, metoda vnitřních bodů apod. V oblasti zaměřené na Rankine-Clausiův cyklus byl tento cyklus teoreticky popsán jak v základním provedení, tak i v jeho modifikacích, jako jsou nadkritický cyklus a zavedení příhřevu.

Druhá část diplomové práce je věnována příkladům. Cílem této části bylo optimalizování vybraných cyklů. V prvním příkladu byl optimalizován ideální cyklus. Prvním krokem k nalezení řešení bylo sestavení matematického modelu. Dále se pokračovalo na implementaci modelu do programu MATLAB (popsáný postup byl použit u všech příkladů). Tato implementace byla popsána jak ze stránky syntaxe (pomocí textu), tak i ze stránky strukturální (pomocí vývojových diagramů). Výsledkem byla konfigurace s termickou účinností $42,35$ %, pro niž byl vykreslen t-s diagram. Druhým příkaldem byla citlivostní analýza ideálního cyklu. Účelem příkladu bylo bližší pochopení chování modelu. Porovnávali jsme výsledné hodnoty termické účinnosti pří nastavení pevných hodnto proměnných. Z této analýzy nám vyšly tři závislosti. První byla spjatá s velikostí tlaku ohřevu a určovala bod obratu nárůstu termické účinnosti, kdy po tomto bodu termická účinnost klesá. Příčinou obratu je podmínka minimální suchostí na výstupu z turbíny. Tato podmínka určovala i další závislosti, jako závislost mezi maximální dovolenou teplotou přehřáté páry a tlakem ohřevu a snižováním termické účinnosti při zvětšování tlaku kondenzace. Po citlivostní analýze jsme se zabývali nadkritickým cyklem. Pro tento cyklus byl pozměněn matematický model prvního příkladu. Výsledkem byla termická účinnost $46,41$ %. Poslední modifikací základní úlohy bylo přidání příhřevu. Tento problém byl popsán v příkladu tři a čtyři, přičemž třetí příklad se zabýval ideální konfigurací Rankine-Clausiova cyklu s jedním příhřevem a čtvrtý s více přihřevy. Příklad s jedním příhřevem sloužil pro sestavení matematického modelu a názornější ukázku výpočtové struktury. Výsledkem byla konfigurace s termickou účinnosti $45,17$ %. Poslední příklad byl o poznání složitější, a proto se rozdělil dále na dvě dílčí úlohy. První z nich se zabývala ideální velikostí jednotlivých příhřevů a druhá počtem příhřevů. Výsledkem byla konfigurace se čtyřmi příhřevy a s konstantní velikostí příhřevu, přičemž její termická účinnost byla $47,87$ %.

Zadané cíle práce, zmíněné v úvodu, byly splněny. A jelikož se tato práce zabývá ideálním Rankine-Clausiovým cyklem, tak jako další možné pokračování je zavedení neideálnosti cyklu, anebo zavedení pravděpodobnosti do dějů.

Literatura

[1] BAZARAA, M, Hanif D SHERALI a C SHETTY. *Nonlinear programming: theory and algorithms*. 3rd ed. Hoboken: John Wiley & Sons, 2006, xv, 853 s. ISBN 0-471-48600-0.

[2] BAZARAA, M, , John J JARVIS a Hanif D SHERALI. *Linear programming and network flows*. 4th ed. Hoboken: John Wiley & Sons, 2010, xiv, 748 s. ISBN 978-0-470-46272-0.

[3] ČAJANEK, M. *Modely stochastického programování v inženýrském návrhu*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2009, 44s.

[4] ÇENGEL, Yunus A a Michael A BOLES. *Thermodynamics: an engineering approach*. 7th ed. in SI units. New York: McGraw-Hill, c2011, xxv, 978 s. ISBN 978-007-131111-3.

[5] MAREŠ, T.

Konstrukční optimalizace. Vyd. 1. Praha : Nakladatelství ČVUT, 2007, 106s. ISBN 978-80-01-03696-9.

[6] THE MATHWORKS, Inc. MathWorks: Accelerating the pace of engineering and science [online]. 2013 [cit. 2013-05-11]. Dostupné z: http://www.mathworks.com/index.html

[7] POPELA, P. *Nonlinear programming*. 2002, 72s.

[8] WAGNER, Wolfgang a Hans-Joachim KRETZSCHMAR. *International Steam Tables: Properties of Water and Steam based on the Industrial Formulation IAPWS-IF97*. 2nd ed. Berlin: Springer, 2008. ISBN 978-3-540-21419-9.

[9] X Steam: Thermodynamic properties of water and steam. THE MATHWORKS, Inc. Matlab central [online]. 2006, 01.08.2007 [cit. 2013-05-11]. Dostupné z: http://www.mathworks.com/matlabcentral/fileexchange/9817

Seznam použitých zkratek, symbolů a veličin

Symbol	Jednotka	Veličina
ds		Hodnota přesnosti vykreslování
$\Re$		Množina všech reálných čísel
R-C cyklus		Rankine-Clausiův cyklus
S		Konvexní množina
$\mathbf{X}$		Množina přípustných řešení
$\mathbf{x}$		Vektor proměnných
a	$[kJ/kg]$	Měrná práce
d	$[kg/h]$	Výrobní kapacita uhlí
e	$[kg/h]$	Spotřeba uhlí
i	$[kJ/kg]$	Měrná entalpie
p	$[bar]$	Tlak
q	$[kJ/kg]$	Měrné teplo
s	$[kJ/kgK]$	Měrná entropie
t	$[°C]$	Teplota
x	$[-]$	Suchost páry
η_t	$[\%]$	Termická účinnost
$_{č}$		Index pro čerpadlo
$_{in}$		Index pro vstup
$_{min}$		Index pro minimum
$_{out}$		Index pro výstup
$_{T}$		Index pro turbínu
$_{0}$		Index pro počáteční stav
$_{'}$		Index pro sytou kapalinu

Seznam příloh

A Kód pro distribuční úlohu **45**

B Kód pro ideální cyklus **46**

C Kód pro citlivostí analýzu ideálního cyklu **49**

D Kód pro cyklus s jedním příhřevem **51**

E Kód pro cyklus s více přehřevy **54**
E.1 Kód pro optimalizaci velikosti příhřevu 57
E.2 Kód pro optimalizaci počtu příhřevů 58

A. Kód pro distribuční úlohu (kapitola 2.1.3)

```
    clc; clear;
% výpočet distribuce pomocí simplexové metody
% vektor proměnných x=(x1,x2,x3,x4,x5,x6,s1,s2)
% počáteční bod algoritmu
x0=[0;0;0;0;0;0;0;0];
% nastavení minimální a maximalní hodnoty
hor=[inf;inf;inf;inf;inf;inf;inf;inf];
dol=[0;0;0;0;0;0;0;0];
% nastavení podmínkových funkcí ve tvaru rovnic Aeq*x = Beq
Aeq=[1,0,0,1,0,0,0,0
0,1,0,0,1,0,0,0
0,0,1,0,0,1,0,0
1,1,1,0,0,0,1,0
0,0,0,1,1,1,0,1];
Beq=[325;275;300;600;350];
% nastavení účelové funkce f*x
f=[25;14;18;25;18;17;0;0];
% nastavení výběru metody řešení
options = optimset('LargeScale', 'off', 'Simplex', 'on');
[x, fval] = linprog(f,[],[],Aeq,Beq,dol,[],x0,options);
```

B. Kód pro ideální cyklus (kapitola 4.1)

```
    % vektor neznámých
% n(n1,n2,n3)
clc;clear;
% nastavení přesnosti vykreslování grafu
ds=1000;
% nastavení použitého algoritmu a hodnot pro ukončení řešiče
options = optimset('Algorithm','interior-point');
options.MaxFunEvals=10000;
% nastavení počátečního bodu
n0=[30;200;15];
% nastavení horních a dolních vektorů
hormez=[170;565;inf];
dolmez=[0;0;0.08];
% nadkritický cyklus
% hormez=[inf;700;inf];
% pomocná proměnná na počítání počtu cyklu
p=0;
while true
% řešení úlohy
% výstupní hodnoty
% n-výsledná konfigurace
% fval-výsledná hodnota účelové funkce, v %, ale záporná max->min
% hodnota-pomocná hodnota určující, zdali je výsledek přípustný
[n, fval, hodnota] = ...
fmincon(@ucelfunkce,n0,[],[],[],[],dolmez,hormez,@omezfunkce,options);
% rozhodování zdali je výsledek přípustný
if (hodnota == 1);
% vykreslování t-s diagramu
graf(n,ds);
break;
else
if (p>10)
break;end
end
n0=n0+[10;0;0];
p=p+1;
end

function f = ucelfunkce(n)
% Účelová funkce
% entropie
s3=XSteam('s_pT',n(1),n(2));
```

```
s1=XSteam('sl_p',n(3));
% entalpie
i2=XSteam('h_ps',n(1),s1);
i3=XSteam('h_pt',n(1),n(2));
i4=XSteam('h_ps',n(3),s3);
i1=XSteam('hL_p',n(3));
% účelová funkce
f =((i3-i4-i2+i1)*100)/(i2-i3);
end

function [c, ceq] = omezfunkce(n)
% Omezující funkce
s3=XSteam('s_pT',n(1),n(2));
x4=XSteam('x_ps',n(3),s3);
% omezení ve tvaru c(x)=<0
c=0.8-x4;
s_1=XSteam('sL_p',n(3));
x_1=XSteam('x_ps',n(3),s_1);
% omezení ve tvaru ceq=0
ceq=0-x_1;
end

function [] = graf(n,ds)
% na vykreslení t-s diagramu
% napočítání bodů pro křivku sytosti
[Sx0 Sx1 Tx]=bodyx(ds);
% entropie
s1=XSteam('sl_p',n(3));
s3=XSteam('s_pT',n(1),n(2));
% pomocný vektor entropie
S=linspace(s1, s3, ds) ;
velikostS=length(S);
% konstantní tlaky p3=n(1) a p1=n(3)
for i=1:1:velikostS
Tp3(i)=XSteam('t_ps',n(1),S(i));
Tp1(i)=XSteam('t_ps',n(3),S(i));
end
Tp1(1)=XSteam('Tsat_p',n(3));
% konstatní entropie
Ts1=[XSteam('Tsat_p',n(3));XSteam('t_ps',n(1),s1)];
Ts3=[XSteam('t_ps',n(1),s3);XSteam('t_ps',n(3),s3)];
% pomocné vektory entropií
Ss1=[s1;s1];
Ss3=[s3;s3];
% vykreslování
plot(Sx0,Tx,'b',Sx1,Tx,'b',S,Tp1,'r',S,Tp3,'r',Ss1,Ts1,'g',Ss3,Ts3,'g',Ss1(1),Ts1(1),'k.',...
Ss1(1),Ts1(2),'k.',Ss3(1),Ts3(1),'k.',Ss3(1),Ts3(2),'k.','LineWidth',2,'MarkerSize',20);
```

```
% vykreslení názvu os
xlabel('s [ kJ/kgK]');
ylabel('t [°C]');
% očíslování bodů
text(Ss1(1),Ts1(1)-25,'1');
text(Ss1(1),Ts1(2)+25,'2');
text(Ss3(1),Ts3(1)+25,'3');
text(Ss3(1),Ts3(2)-25,'4');
end
```

Používá se ve všech zdrojových kódech nezměněna, proto se vypisuje jen zde.

```
function [Sx0,Sx1,Tx] = bodyx(dT)
% napočítání bodů pro křivku sytosti
% dT=ds
% pomocný vektor teplot
Tx=linspace(0, 373.9,dT) ;
velikost=length(Tx);
% křivka syté kapaliny
for i=1:1:velikost
Sx0(i)=XSteam('sL_T',Tx(i));
end
% křivka syté páry
for i=1:1:velikost
Sx1(i)=XSteam('sV_T',Tx(i));
end
% zaručení protnutí
Sx1(velikost) = Sx0(velikost);
end
```

C. Kód pro citlivostí analýzu ideálního cyklu (kapitola 4.2)

```
    % vektor neznámých
% n(n1,n2,n3)
clc;clear;
% nastavení použitého algoritmu a hodnot pro ukončení řešiče
options = optimset('Algorithm','interior-point');
options.MaxFunEvals=10000;
% horní a dolní meze
hmez=[170;565;10];
dmez=[0;0;0.08];
% pomocná proměnná na počítání počtu cyklů
j=1;
% citlivostní analýza na n1
for i=30:10:170
% nastavení počátečního bodu
n0=[i;500;1];
% omezení ve formě lineární funkce
Aeq=[1,0,0];
Beq=i;
% řešení úlohy
% výstupní hodnoty
% np(j,:)-výsledná konfigurace
% fvalp(j)-výsledná hodnota účelové funkce, v %, ale záporná max->min
% hodnota_p(j)-pomocná hodnota určující, zdali je výsledek přípustný
[np(j,:),fvalp(j),hodnota_p(j)] =...
fmincon(@ucelfunkce,n0,[],[],Aeq,Beq,dmez,hmez,@omezfunkce,options);
j=j+1;
end
% zápis výsledků do jedné matice
np(:,4)=-fvalp(:);
np(:,5)=hodnota_p(:);
j=1;
% citlivostní analýza na n2
for i=400:5:565
n0=[170;i;1];
Aeq=[0,1,0];
Beq=i;
[nt(j,:),fvalt(j),hodnota_t(j)] = ...
fmincon(@ucelfunkce,n0,[],[],Aeq,Beq,dmez,hmez,@omezfunkce,options);
j=j+1;
end
nt(:,4)=-fvalt(:);
nt(:,5)=hodnota_t(:);
j=1;
```

```
% citlivostní analýza na n3
for i=0.08:0.01:1
n0=[160;500;i];
Aeq=[0,0,1];
Beq=i;
[npk(j,:), fvalpk(j), hodnota_k(j)] =...
fmincon(@ucelfunkce,n0,[],[],Aeq,Beq,dmez,hmez,@omezfunkce,options);
j=j+1;
end
npk(:,4)=-fvalpk(:);
npk(:,5)=hodnota_k(:);

function f = ucelfunkce(n)
% Účelová funkce
% entropie
s3=XSteam('s_pT',n(1),n(2));
s1=XSteam('sl_p',n(3));
% entalpie
i2=XSteam('h_ps',n(1),s1);
i3=XSteam('h_pt',n(1),n(2));
i4=XSteam('h_ps',n(3),s3);
i1=XSteam('hL_p',n(3));
% účelová funkce
f =((i3-i4-i2+i1)*100)/(i2-i3);
end

function [c, ceq] = omezfunkce(n)
% Omezující funkce
s3=XSteam('s_pT',n(1),n(2));
x4=XSteam('x_ps',n(3),s3);
% omezení ve tvaru c(x)=<0
c=0.8-x4;
s_1=XSteam('sL_p',n(3));
x_1=XSteam('x_ps',n(3),s_1);
% omezení ve tvaru ceq=0
ceq=0-x_1;
end
```

D. Kód pro cyklus s jedním příhřevem (kapitola 4.4)

```
    % vektor neznámých
% n(n1,n2,n3,n4,n5)
clc;clear;
% nastavení přesnosti vykreslování grafu
ds=1000;
% nastavení použitého algoritmu a hodnot pro ukončení řešiče
options = optimset('Algorithm','interior-point');
options.MaxFunEvals=10000;
% nastavení počátečního bodu
n0=[100;565;50;565;1];
% nastavení horních a dolních vektorů
hormez=[170;565;170;565;inf];
dolmez=[0;0;0;0;0.08];
% pomocná proměnná na počítání počtu cyklů
p=0;
while true
% řešení úlohy
% výstupní hodnoty
% n-výsledná konfigurace
% fval-výsledná hodnota účelové funkce, v %, ale záporná max->min
% hodnota-pomocná hodnota určující, zdali je výsledek přípustný
[n, fval, hodnota] = ...
fmincon(@ucelfunkce,n0,[],[],[],[],dolmez,hormez,@omezfunkce,options);
% rozhodování zdali je výsledek přípustný
if (hodnota == 1);
% vykreslování t-s diagramu
graf(n,ds);
break;
else
if (p>10)
break;end
end
n0=n0+[10;0;10;0;0];
p=p+1;
end

function [f] = ucelfunkce(n)
% Účelová funkce
% entropie
s1=XSteam('sl_p',n(5));
s3=XSteam('s_pT',n(1),n(2));
s5=XSteam('s_pT',n(3),n(4));
% entalpie
```

```
i1=XSteam('hL_p',n(5));
i2=XSteam('h_ps',n(1),s1);
i3=XSteam('h_pt',n(1),n(2));
i4=XSteam('h_ps',n(3),s3);
i5=XSteam('h_pt',n(3),n(4));
i6=XSteam('h_pt',n(5),s5);
% účelová funkce
f=((i3-i4+i5-i6+i1-i2)*100)/(i2-i3+i4-i5);
end

function [c, ceq] = omezfunkce(n)
% Omezující funkce
s3=XSteam('s_pT',n(1),n(2));
x4=XSteam('x_ps',n(3),s3);
s5=XSteam('s_pT',n(3),n(4));
x6=XSteam('x_ps',n(5),s5);
% omezení ve tvaru c(x)=<0
c(1)=0.8-x4;
c(2)=0.8-x6;
s1=XSteam('sL_p',n(5));
x1=XSteam('x_ps',n(5),s1);
% omezení ve tvaru ceq=0
ceq=0-x1;
end

function [ ] = graf(n,ds)
% na vykreslení t-s diagramu
% napočítání bodů pro křivku sytosti
[Sx0Sx1Tx]=bodyx(ds);
% entropie
s1=XSteam('sl_p',n(5));
s3=XSteam('s_pT',n(1),n(2));
s5=XSteam('s_pT',n(3),n(4));
% pomocné vektory entropií
S3=linspace(s1, s3, ds) ;
S5=linspace(s3, s5, ds) ;
SK=linspace(s1, s5, ds) ;
velikostS3=length(S3);
velikostS5=length(S5);
velikostSK=length(SK);
% konstantní tlaky p3=n(1)
for i=1:1:velikostS3
Tp3(i)=XSteam('t_ps',n(1),S3(i));
end
% konstantní tlaky p5=p4=n(3)
for i=1:1:velikostS5
Tp5(i)=XSteam('t_ps',n(3),S5(i));
```

```
end
% konstantní tlaky pk=n(5)
for i=1:1:velikostSK
Tpk(i)=XSteam('t_ps',n(5),SK(i));
end
% konstatní entropie
Ts1=[XSteam('Tsat_p',n(5));XSteam('t_ps',n(1),s1)];
Ts3=[XSteam('t_ps',n(1),s3);XSteam('t_ps',n(3),s3)];
Ts5=[XSteam('t_ps',n(3),s5);XSteam('t_ps',n(5),s5)];
% pomocné vektory entropií
Ss1=[s1;s1];
Ss3=[s3;s3];
Ss5=[s5;s5];
% vykreslování
plot(Sx0,Tx,'b',Sx1,Tx,'b',Ss1,Ts1,'g',Ss3,Ts3,'g',Ss5,Ts5,'g',...
S3,Tp3,'r',SK,Tpk,'r',S5,Tp5,'r',Ss1(1),Ts1(1),'k.',Ss1(1),...
Ts1(2),'k.',Ss3(1),Ts3(1),'k.',Ss3(1),Ts3(2),'k.',Ss5(1),Ts5(1),'k.',...
Ss5(1),Ts5(2),'k.','LineWidth',2,'MarkerSize',20);
% vykreslení názvu os
xlabel('s [ kJ/kgK]');
ylabel('t [°C]');
% očíslování bodů
text(Ss1(1),Ts1(1)-25,'1');
text(Ss1(1),Ts1(2)+25,'2');
text(Ss3(1),Ts3(1)+25,'3');
text(Ss3(1),Ts3(2)-25,'4');
text(Ss5(1),Ts5(1)+25,'5');
text(Ss5(1),Ts5(2)-25,'6');
end
```

E. Kód pro cyklus s více přehřevy

Funkce společné pro oba případy:

```
function [f] = ucelfunkce( n )
% Účelová funkce
% počet příhřevů
global k;
% entropie s1,s2,s3,...
S(1)=XSteam('sl_p',n(1));
j=3;
for i=2:2:2*k+2
S(j)=XSteam('s_pT',n(i),n(i+1));
j=j+2;
end;
% entalpie
I(1)=XSteam('hL_p',n(1));
j=2;
for i=2:2:2*k+2
I(j)=XSteam('h_ps',n(i),S(i-1));
I(j+1)=XSteam('h_pt',n(i),n(i+1));
j=j+2;
end;
I(j)=XSteam('h_ps',n(1),S(end));
% účelová funkce
f1=0;
f2=0;
for i=3:2:length(I)-1
f1=f1+I(i)-I(i+1);
end
f1=f1+I(1)-I(2);
for i=2:2:length(I)-2
f2=f2+I(i+1)-I(i);
end
f=(f1*100)/(-f2);
end
```

```
function [ ] = graf(n,ds)
% na vykreslení t-s diagramu
% napočítání bodů pro křivku sytosti
[Sx0 Sx1 Tx]=bodyx(ds);
% počet příhřevů
global k;
% entropie s1,s2,s3,...
s(1)=XSteam('sl_p',n(1));
j=2;
for i=2:2:2*k+2
```

```
s(j)=XSteam('s_pT',n(i),n(i+1));
j=j+1;
end;
% generátor množin
j=1;
S=zeros(k+2,ds);
% pomocné vektory entropií, zapsané maticově
for i=1:1:k+1
S(j,:)=linspace(s(i), s(i+1), ds);
j=j+1;
end
S(j,:)=linspace(s(1), s(end), ds);
% generátor T podle matice S, konstantní tlaky
j=1;
for l=2:2:2*k+2
for i=1:1:ds
Tp(j,i)=XSteam('t_ps',n(l),S(j,i));
end
j=j+1;
end
Tp(j,1)=XSteam('Tsat_p',n(1));
for i=2:1:ds
Tp(j,i)=XSteam('t_ps',n(1),S(j,i));
end
% generátor T podle matice S, konstantní entropie
Ts=zeros(k+2,2);
Ts(1,1)=XSteam('Tsat_p',n(1));
Ts(1,2)=XSteam('t_ps',n(2),s(1));
l=2;
for j=2:1:k+2
Ts(j,1)=XSteam('t_ps',n(l),s(j));
if l==2*k+2
l=-1;
end
Ts(j,2)=XSteam('t_ps',n(l+2),s(j));
l=l+2;
end
% pomocné vektory entropií
for j=1:1:k+2
Ss(j,1)=s(j);
Ss(j,2)=s(j);
end
figure;
hold on
% vykreslení meze sytosti
plot(Sx0,Tx,'b',Sx1,Tx,'b','LineWidth',2);
% vykreslení izobar
```

```
for j=1:1:k+2
plot(S(j,:),Tp(j,:),'r','LineWidth',2)
end
% vykreslení izoentrop
for j=1:1:k+2
plot(Ss(j,:),Ts(j,:),'g','LineWidth',2)
end
% vykreslení bodů
for j=1:1:k+2
plot(s(j),Ts(j,1),'k.','MarkerSize',20)
plot(s(j),Ts(j,2),'k.','MarkerSize',20)
end
% očíslování bodů
text(s(1),Ts(1,2)+25,'2');
text(s(1),Ts(1,1)-25,'1');
i=3;
for j=2:1:k+2
text(s(j),Ts(j,1)+25,num2str(i));
text(s(j),Ts(j,2)-25,num2str(i+1));
i=i+2;
end
% vykreslení názvu os
xlabel('s [ kJ/kgK]');
ylabel('t [°C]');
hold off
end

function [c, ceq] = omezfunkce( n )
% omezující funkce
% počet příhřevů
global k;
% výpočet entropie
S(1)=XSteam('sl_p',n(1));
j=3;
for i=2:2:2*k+2
S(j)=XSteam('s_pT',n(i),n(i+1));
j=j+2;
end;
% výpočet entalpie
I(1)=XSteam('hL_p',n(1));
j=2;
for i=2:2:2*k+2
I(j)=XSteam('h_ps',n(i),S(i-1));
I(j+1)=XSteam('h_pt',n(i),n(i+1));
j=j+2;
end;
I(j)=XSteam('h_ps',n(1),S(end));
```

```
j=1;
% výpočet jednotlivých prací
for i=3:2:length(I)-1
a(1,j)=I(i)-I(i+1);
j=j+1;
end
% omezení ve tvaru c(x)=<0
x_end=XSteam('x_ps',n(1),S(end));
c=0.85-x_end;
% omezení ve tvaru ceq=0
x_1=XSteam('x_ps',n(1),S(1));
ceq(1)=0-x_1;
% omezení stejné práce, když nepoužijeme zakomentujeme
for j=2:1:length(a)-1
ceq(j)=a(1,j)-a(1,j-1);
end
end
```

E.1. Kód pro optimalizaci velikosti příhřevu (kapitola 4.5.1)

```
% vektor neznámých
% n(n1,n2,n3,n4,n5,...)
clc;clear;
% nastavení přesnosti vykreslování grafu
ds=1000;
% nastavení použitého algoritmu a hodnot pro ukončení řešiče
options = optimset('Algorithm','interior-point');
options.MaxFunEvals = 100000;
% počet příhřevů
global k;
k=4;
% nastavení počátečního bodu
n0(1)=0.08;
for i=2:2:2*k+2
n0(i)=170-i*10;
n0(i+1)=565;
end
% nastavení horních a dolních vektorů
hormez(1)=inf;
dolmez(1)=0.08;
for i=2:2:2*k+2
hormez(i)=170;
hormez(i+1)=565;
dolmez(i)=0;
```

```
dolmez(i+1)=0;
end
p=1;
while true
% řešení úlohy
% výstupní hodnoty
% n-výsledná konfigurace
% fval-výsledná hodnota účelové funkce, v %, ale záporná max->min
% hodnota-pomocná hodnota určující, zdali je výsledek přípustný
[n, fval, hodnota] = ...
fmincon(@ucelfunkce,n0,[],[],[],[],dolmez,hormez,@omezfunkce,options);
% rozhodování, zdali je výsledek přípustný
if (hodnota == 1);
graf(n,ds);
break;
else
if (p>10)
break;end
end
for i=2:2:2*k+2
n0(i)=n0(i)+10;
end
p=p+1;
end
```

E.2. Kód pro optimalizaci počtu příhřevů (kapitola 4.5.2)

```
% vektor neznámých
% n(n1,n2,n3,n4,n5,...)
clc;clear;
% nastavení přesnosti vykreslování grafu
ds=1000;
% nastavení použitého algoritmu a hodnot pro ukončení řešiče
options = optimset('Algorithm','interior-point');
options.MaxFunEvals=10000;
% počet příhřevů
global k;
% nastavení počátečního bodu
for k=1:1:25
n0(1)=0.08;
for i=2:2:2*k+2
n0(i)=170-i*10;
n0(i+1)=565;
end
% nastavení horních a dolních vektorů
```

```
hormez(1)=inf;
dolmez(1)=0.08;
for i=2:2:2*k+2
hormez(i)=170;
hormez(i+1)=565;
dolmez(i)=0;
dolmez(i+1)=0;
end
p=1;
while true
% řešení úlohy
% výstupní hodnoty
% n-výsledná konfigurace
% fval-výsledná hodnota účelové funkce, v %, ale záporná max->min
% hodnota-pomocná hodnota určující, zdali je výsledek přípustný
[n, fval, hodnota] = ...
fmincon(@ucelfunkce,n0,[],[],[],[],dolmez,hormez,@omezfunkce,options);
% rozhodování, zdali je výsledek přípustný
if (hodnota == 1);
% vykreslování-jen na vyhodnocení správnosti výsledků
graf(n,ds);
break;
else
if (p>100)
break;end
end
for i=2:2:2*k+2
n0(i)=n0(i)+10;
end
p=p+1;
end
% zapsání jednotlivých výsledků
vysledek(k,1)=-fval;
vysledek(k,2)=hodnota;
end
```