



Dokumen Pengembangan TRIAC **(TRIso *Analysis Code*)**

LABORATORIUM KOMPUTASI
PUSAT TEKNOLOGI DAN KESELAMATAN REAKTOR NUKLIR

Disusun oleh:
Arya Adhyaksa Waskita

Supervisor:
Dr. Topan Setiadipura

31 Juli 2017

Daftar Isi

Daftar Gambar	ii
Daftar Program	iii
1 Pendahuluan	2
2 Alur Perhitungan	4
2.1 Pendahuluan	4
2.2 Membaca <i>file input</i>	5
2.3 Menghitung OPF saat irradiasi	6
2.4 Menghitung DS saat kecelakaan	6
2.5 Menghitung tekanan	7
2.6 Fraksi gagal bahan bakar	8
2.6.1 Fraksi gagal akibat berkurangnya <i>tensile strength</i>	8
2.6.2 Fraksi gagal bahan bakar akibat <i>weight loss</i>	9
3 Penerapan	11
3.1 Pendahuluan	11
3.2 Saat irradiasi	15
LAMPIRAN	1
Lampiran 1	2

Daftar Gambar

1.1	Ilustrasi bentuk bahan bakar <i>pebble</i>	2
1.2	Komposisi elemen pelapis partikel	3
2.1	Diagram alir perhitungan TRAIC	4

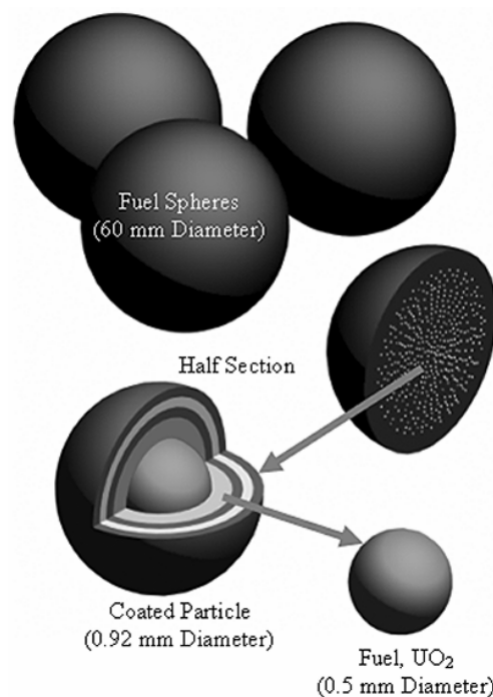
Daftar Program

3.1	triac.py	11
-----	--------------------	----

BAB 1

Pendahuluan

BATAN saat ini tengah berencana membangun reaktor riset baru berbasis HTGR (*High Temperature Gas-cooled Reactor*) [1] sebagai persiapan PLTN, yang akan dibangun di Indonesia di masa depan [2]. Salah satu yang perlu diperhatikan dalam pengembangan reaktor jenis ini adalah bahan bakarnya yang berjenis *pebble* yang bentuknya dapat diilustrasikan seperti pada Gambar 1.1. Bahan bakar harus dirancang sedemikian rupa sehingga rasio gagalnya bahan bakar selama operasi minimal.

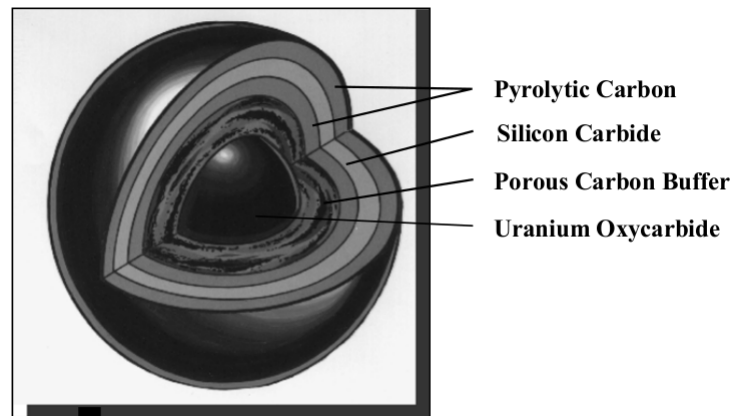


Gambar 1.1: Ilustrasi bentuk bahan bakar *pebble* [1]

Bahan bakar berjenis *pebble* ini memiliki komponen utama yang dalam Gambar 1.1 disebut sebagai *coated particle*. Komposisi elemen pelapis (*coated*) dapat diilustrasikan dalam Gambar 1.2. Dalam upaya menguasai teknologi reaktor berjenis HTGR melalui pengembangan RDE, salah tugas yang harus dilaksanakan adalah penguasaan analisis kegagalan bahan bakarnya, khususnya ketika terjadi kecelakaan.

Beragam model analisis telah dikembangkan, salah satunya yang dikembangkan oleh Wang [1]. Selain itu, terdapat sebuah model sederhana yang dikembangkan oleh Verfondern dalam PANAMA [3]. Pada model tersebut, bahan bakar disebut gagal jika kekuatan lapisan

SiC (*Silicon Carbide*) lebih kecil daripada tekanan internal dari lapisan di bawahnya (perhatikan Gambar 1.2). Model inilah yang akan diterapkan dalam TRIAC (*TRIso Analysis Code*).



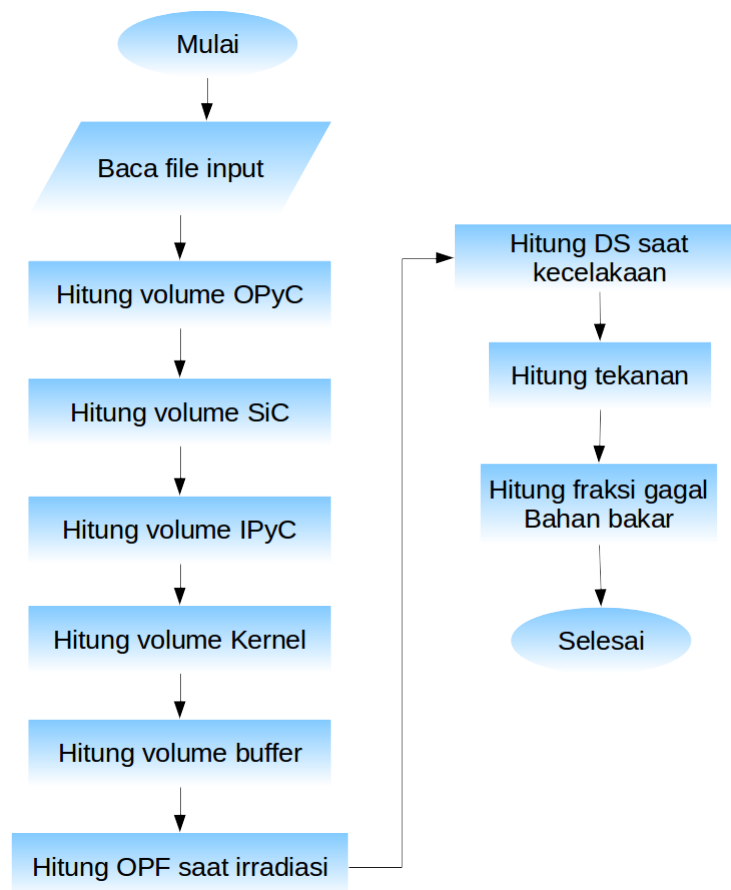
Gambar 1.2: Komposisi elemen pelapis partikel [1]

BAB 2

Alur Perhitungan

2.1 Pendahuluan

Secara umum, perhitungan TRIAC mengikuti diagram alir seperti pada Gambar 2.1 berikut. Sementara kode sumbernya disajikan dalam Listing 3.1 yang dibangun sepenuhnya berbasis pengetahuan yang diperoleh dari dokumen laporan teknis [4].



Gambar 2.1: Diagram alir perhitungan TRIAC

2.2 Membaca *file input*

Sub rutin ini ditujukan untuk membaca file input dengan format seperti terdapat pada Lampiran 1. Sub rutin ini menggunakan skema yang kaku karena identifikasi nilai-nilai yang akan dibaca ditentukan oleh suatu teks tertentu. Setelah teks yang menjadi penanda, nilai-nilai yang dibutuhkan dibaca. Tetapi, nilai tersebut dapat langsung berada dalam satu baris bersama dengan teks penanda, atau berada pada baris yang berbeda. Sub rutin ini terdapat pada baris ke-3 s/d baris ke-69 dalam Listing 3.1

Terdapat empat jenis data yang perlu dibaca dari *file input* dalam Lampiran 1, masing-masing adalah sebagai berikut.

1. Data tentang geometri *pebble*. Data ini diidentifikasi menggunakan teks yang didefinisikan oleh variabel `statusGeometry` (baris ke-5 pada Listing 3.1). Di dalam data geometri, terdapat empat data berbeda, masing-masing secara berurutan adalah panjang jejari *pebble* terluar, OPyC (*Outer Pyrolytic Carbon*), SiC (*Silicon Carbide*), IPyC (*Inner Pyrolytic Carbon*), *buffer* dan *kernel*. Data geometri akan digunakan untuk menghitung volume setiap elemen pelapis (Gambar 1.2). Yang perlu diperhatikan adalah data jari-jari yang disajikan adalah jarak dari pusat bahan bakar sampai titik terluar dari setiap lapisan. Karena itu, volume suatu lapisan harus mempertimbangkan lapisan-lapisan di dalamnya. Data geometri disimpan dalam variabel diberi nama `dimensi` dan dalam bentuk `list` (baris ke-10 dalam Listing 3.1).
2. Data tentang kekuatan SiC. Data ini diidentifikasi menggunakan teks yang didefinisikan oleh variabel `statusCharacteristics` (baris ke-6 pada Listing 3.1). Ada empat nilai yang perlu dibaca terkait kekuatan SiC, masing-masing adalah SiC *Tensile Strength* [Pa], *Weibull Modulus Burnup* [FIMA], *Fission Yield of stable fission gasses* [Ff], *Fast Neutron Fluence* dan rasio berat Th terhadap U-235 pada *kernel*. Data terkait kekuatan SiC disimpan dalam variabel yang diberi nama `characteristics` dalam bentuk `list` (baris ke-11 dalam Listing 3.1).
3. Data tentang sejarah iradiasi. Data ini diidentifikasi menggunakan teks yang didefinisikan oleh variabel `statusIrradiation` (baris ke-7 pada Listing 3.1). Data ini merupakan data temperatur bahan bakar *pebble* pada selang waktu tertentu. Sebagai contoh, data yang disajikan pada Lampiran 1 diambil pada selang waktu 17 hari. Data sejarah iradiasi disimpan dalam variabel yang diberi nama `irradiation` dalam bentuk `list`. Setiap elemen adalah `list` yang secara *nested* terdiri dari dua elemen yang mewakili data kolom kedua dan ketiga tiap akuisisi (baris ke-12 pada Listing 3.1). Ilustrasinya adalah seperti `[[0,593],[17,833],...]`. Data tentang nomor urut tidak digunakan karena selain tidak diperlukan dalam perhitungan, akan menyulitkan proses interpolasi yang akan diterapkan berikutnya.
4. Data tentang sejarah kecelakaan. Data ini diidentifikasi menggunakan teks yang didefinisikan oleh variabel `statusAccident` (baris ke-8 pada Listing 3.1). Data ini memiliki pola yang sama dengan data sejarah iradiasi. Data sejarah kecelakaan disimpan dengan cara yang sama seperti data tentang sejarah iradiasi tetapi dengan nama `accident` (baris ke-13 pada Listing 3.1). Ilustrasinya adalah seperti `[[0,1033],[0.0271,1033],...]`

2.3 Menghitung OPF saat irradiasi

OPF (*Oxygen Per Fission*) adalah jumlah atom oksigen yang terlepas selama fisi atom U^{235} atau Pu^{239} . Atom oksigen ini mempengaruhi terbentuknya senyawa CO yang akan meningkatkan tekanan internal dalam bahan bakar. Pembentukan senyawa CO juga dipengaruhi oleh temperatur, waktu serta jenis partikel kernel.

Nilai OPF didekati oleh persamaan (2.1). Nilai n dalam persamaan (2.1) sama dengan banyaknya data sejarah irradiasi. Nilai Δ_i merupakan selisih waktu dari sejarah irradiasi yang dicatat. Nilainya akan berubah dengan berubahnya rentang pencatatan temperatur irradiasi. Jika dalam contoh kasus yang disajikan pada Lampiran 1, rentang waktu pencatatan temperatur irradiasi dilakukan setiap 17 hari, maka Δ_i adalah 17 hari atau $17 \times 24 \times 3600$ detik. t_B adalah waktu irradiasi total bahan bakar, sedangkan \bar{t}_i waktu irradiasi ketika pencatatan dilakukan.

$$OPF \simeq \sum_{i=1}^n g(\bar{t}_i) \cdot (t_B - \bar{t}_i) \cdot \Delta t_i \quad (2.1)$$

Tetapi, nilai OPF juga didefinisikan seperti persamaan (2.2), dengan nilai $g(\bar{t}_i)$ didefinisikan oleh persamaan (2.3). Nilai R pada persamaan (2.3) adalah konstanta gas sebesar $8.3143 \left[\frac{J}{mole \cdot K} \right]$.

$$OPF = \frac{g(T)}{2} \cdot t^2 \quad (2.2)$$

$$\frac{g(T)}{2} = 8.32 \cdot 10^{-11} \cdot e^{-\frac{163000}{RT}} \quad (2.3)$$

Nilai OPF selanjutnya digunakan untuk menghitung nilai temperatur irradiasi (T_B) dari persamaan (2.4). Formula empiris tersebut sesuai untuk jenis bahan bakar UO_2 .

$$\log OPF = -10.08 - \frac{0.85 \cdot 10^4}{T_B} + 2 \cdot \log t_B \quad (2.4)$$

Sedangkan nilai T_B akan digunakan untuk menghitung DS , faktor berkurangnya koefisien difusi (s^{-1}) dari gas hasil fisi di dalam partikel kernel. Nilainya untuk bahan bakar UO_2 memenuhi persamaan (2.5).

$$\log DS = -2.30 - \frac{0.8116 \cdot 10^4}{T_B} \quad (2.5)$$

Terakhir, DS akan digunakan untuk menghitung sebuah nilai tak berdimensi τ_i yang memenuhi persamaan (2.6).

$$\tau_i = DS(T_B) \cdot t_B \quad (2.6)$$

2.4 Menghitung DS saat kecelakaan

Seperti telah dijelaskan dalam sub bab 2.3, DS adalah faktor berkurangnya koefisien difusi gas hasil fisi dalam partikel kernel. Sekarang, faktor ini dihitung ketika kondisi kecelakaan terjadi. Kita memerlukan sejarah temperatur bahan bakar setelah kecelakaan terjadi serta τ_i yang telah dihitung di persamaan (2.6).

Dengan menggunakan persamaan (2.6), kita dapat menghitung nilai DS dengan temperatur kecelakaan yang tercatat. Kemudian, kita perlu menghitung nilai τ_A dengan persamaan

(2.6) tetapi dengan nilai temperatur dan waktu setelah terjadi kecelakaan. Selanjutnya, dengan modal nilai τ_i dan τ_A kita akan menghitung nilai Fd , yang merupakan faktor fisi gas Xe dan Kr (yang dominan). Nilai Fd dihitung dengan persamaan (2.7).

$$Fd = \frac{(\tau_i + \tau_A) \cdot f(\tau_i + \tau_A) - \tau_A \cdot f(\tau_A)}{\tau_i} \quad (2.7)$$

Sedangkan nilai $f(\tau)$ dihitung menggunakan persamaan (2.8). Batas atas nilai n pada persamaan (2.8) dapat menggunakan nilai yang cukup besar, misalnya 1000, atau ketika dua nilai berdekatan yang dihasilkan hanya berselisih kurang dari 10^{-20} . Idealnya, suku penjumlahan sebanyak n akan semakin baik jika hasilnya mendekati 1.

$$f(\tau) = 1 - \frac{6}{\tau} \cdot \sum_{n=1}^{\infty} \left(\frac{1 - e^{-n^2 \cdot \pi^2 \cdot \tau}}{n^4 \cdot \pi^4} \right) \quad (2.8)$$

2.5 Menghitung tekanan

Tekanan adalah variabel yang penting dalam tahapan analisis ini karena akan menentukan fraksi gagal bahan bakar. Untuk menghitung tekanan yang timbul ketika kecelakaan terjadi pada waktu tertentu, sehingga menyebabkan panas tertentu, digunakan persamaan (2.9) [1].

$$p = \frac{(F_d \cdot F_f + OPF) \cdot F_b \cdot \left(\frac{V_f}{V_k}\right) \cdot R \cdot T}{V_m} \quad (2.9)$$

dengan :

F_d = fraksi relatif gas fisi yang lepas

F_f = produk fisi yang dihasilkan dari gas fisi stabil, $F_f=0.31$

OPF = jumlah atom oksigen setiap terjadi fisi saat terjadi kecelakaan

F_b = burnup logam berat (FIMA)

V_f = fraksi void [m^3], terkait dengan 50% volume buffer

V_k = volume kernel [m^3]

V_m = volume molar dalam partikel kernel $\left[\frac{m^3}{mole} \right]$

R = konstanta gas, $8.3143 \left[\frac{J}{(mole \cdot K)} \right]$

Khusus untuk variabel OPF , karena perhitungan tekanan dilakukan ketika terjadi kecelakaan, digunakanlah persamaan (2.10). Persamaan (2.10) mirip dengan persamaan (2.4) dengan penambahan suku ke-3.

$$\log OPF = -10.08 - \frac{0.85 \cdot 10^4}{T_B} + 2 \cdot \log t_B - 0.04 \cdot \left(\frac{10^4}{T} + \frac{10^4}{T_B + 75} \right) \quad (2.10)$$

2.6 Fraksi gagal bahan bakar

Tahapan terakhir dari analisis ini adalah perhitungan fraksi gagal bahan bakar. Secara umum, fraksi gagal bahan bakar dipengaruhi sejumlah sebab. Dalam analisis yang dilakukan TRAIC (dan juga PANAMA sebagai acuannya), gagalnya bahan bakar dapat disebabkan oleh 3 sebab. Ketiganya adalah sebagai berikut.

1. Pabrikasi (ϕ_0). Dalam analisis ini, nilai ϕ_0 diasumsikan sama dengan 0.
2. Berkurangnya *tensile strength* lapisan SiC (ϕ_1). Hal ini dapat terjadi karena
 - proses iradiasi maupun
 - meningkatnya temperatur secara signifikan ketika terjadi kecelakaan) atau disebut juga *grain boundary*.
3. Dekomposisi termal pada temperatur tinggi yang menyebabkan terjadinya *weight loss* pada lapisan SiC (ϕ_2).

Ketiga sebab terjadinya kegagalan bahan bakar tersebut mengikuti persamaan (2.11).

$$\phi_{total} = 1 - (1 - \phi_1) - (1 - \phi_2) \quad (2.11)$$

2.6.1 Fraksi gagal akibat berkurangnya *tensile strength*

Nilai fraksi gagal bahan bakar pada waktu t setelah terjadinya kecelakaan diperoleh dengan persamaan (2.12).

$$\phi_1(t, T) = 1 - e^{-\ln 2 \cdot \left(\frac{\sigma_t}{\sigma_o}\right)^m} \quad (2.12)$$

dengan :

σ_o =*tensile strength* dari SiC [Pa] pada akhir iradiasi

σ_t =tekanan yang dialami SiC [Pa] akibat tekanan gas internal

Variabel tekanan internal pada SiC (σ_t) dihitung dengan persamaan (2.13). Pada persamaan (2.13), jari-jari lapisan SiC merupakan rerata karena lapisan SiC memang memiliki ketebalan yang nilai awalnya diwakili oleh variabel d_o .

$$\sigma_t = \frac{r \cdot p}{2 \cdot d_o} \cdot \left(1 + \frac{\dot{v} \cdot t}{d_o}\right) \quad (2.13)$$

dengan :

r =rerata jari-jari SiC, $\left(0.5 \cdot (r_a^3 + r_i^3)\right)^{\frac{1}{3}}$ [m]

d_o =ketebalan awal lapisan SiC, $r_a - r_i$ [m]

p =tekanan gas fisi dalam partikel [Pa]

\dot{v} =laju korosi sebagai fungsi temperatur (T), $\left[\frac{m}{s}\right]$

Sedangkan variabel laju korosi (\dot{v}) dihitung dengan persamaan (2.14), mirip dengan persamaan (2.3) dengan perbedaan pada konstanta.

$$\dot{v} = 5.87 \cdot 10^{-7} \cdot e^{-\left(\frac{179500}{RT}\right)} \quad (2.14)$$

Selanjutnya, variabel *tensile strength* lapisan SiC, penurunan nilainya mengikuti persamaan (2.15). Variabel σ_{oo} merupakan *tensile strength* awal sebelum diiradiasi. Nilainya merupakan sesuatu yang dapat diukur. Sedangkan Γ dan Γ_s masing-masing merupakan *fluence* neutron cepat $[10^{25} m^{-2} EDN]$ dan *fluence* yang dipengaruhi temperatur iradiasi. Nilai Γ_s ditentukan menggunakan persamaan (2.16). Nilai minimum σ_{oo} merupakan nilai awal *tensile strength* dan diasumsikan sama dengan 196 [MPa]. Tentunya, dengan perlakuan iradiasi yang sama, lapisan SiC dengan nilai awal *tensile strength* terkecil akan memiliki nilai akhir *tensile strength* yang juga kecil.

$$\sigma_o = \sigma_{oo} \cdot \left(1 - \frac{\Gamma}{\Gamma_s}\right) \quad (2.15)$$

$$\log \Gamma_s = 0.556 + \frac{0.065 \cdot 10^4}{T_B} \quad (2.16)$$

Tensile strength lapisan SiC yang dihitung menggunakan persamaan (2.15) merupakan nilai yang berlaku pada satu *coated particle*. Padahal, ada sangat banyak *coated particle* yang dioperasikan. Karena itu, diperlukan perhitungan yang mempertimbangkan variabel ini untuk semua distribusi *coated particles*. Dengan pendekatan yang sama seperti persamaan (2.15), persamaan (2.17) dibangun.

$$m_o = m_{oo} \cdot \left(1 - \frac{\Gamma}{\Gamma_m}\right) \quad (2.17)$$

dengan $\log \Gamma_m = 0.394 + \frac{0.065 \cdot 10^4}{T_B}$ dan nilai $m_{oo} = 2$ sebagai nilai terkecilnya. Nilai m_o pada persamaan (2.17) kemudian akan disubstitusi ke persamaan (2.12) sebagai m .

Selain korosi karena proses iradiasi, lapisan SiC juga dapat terkorosi karena *grain Boundary*. Jika korosi akibat iradiasi tergantung pada sejarah iradiasi yang dialami bahan bakar dan terjadi sebelum kecelakaan, maka korosi karena *grain Boundary* terjadi setelah kecelakaan. Penurunan nilai distribusi *tensile strength* akibat meningkatnya temperatur karena kecelakaan mengikuti persamaan (2.18).

$$m = m_o \cdot \left(0.44 + 0.56 \cdot e^{-\dot{\eta} \cdot t}\right) \quad (2.18)$$

di mana nilai $\dot{\eta}$ mengikuti persamaan 2.19 dengan pola yang sama seperti persamaan (2.14).

$$\dot{\eta} = 0.565 \cdot e^{\left(\frac{-187400}{R \cdot T}\right)} [s^{-1}] \quad (2.19)$$

2.6.2 Fraksi gagal bahan bakar akibat *weight loss*

Laju *weight loss* yang terjadi akibat tingginya temperatur saat terjadi kecelakaan mengikuti persamaan (2.20).

$$k = k_o \cdot e^{\frac{-Q}{R \cdot T}} \quad (2.20)$$

dengan $Q = 556 \left[\frac{kJ}{mol}\right]$ dan k_o adalah faktor frekuensi yang tergantung pada jenis partikel.

Selanjutnya, diasumsikan bahwa partikel TRISO tergantung pada apa yang disebut sebagai "*action integral*", dan disimbolkan dengan ζ yang nilainya mengikuti persamaan (2.21).

$$\zeta = \int_{t_1}^{t_2} k(T) dt \quad (2.21)$$

dengan $K(T)$ adalah nilai yang menggambarkan sejarah kondisi partikel yang bergantung pada temperatur dan waktu.

Secara numerik, persamaan (2.21) dapat dituliskan sebagai persamaa (2.22).

$$\zeta(t_2) = \zeta(t_1) + k(T_m) \cdot (t_2 - t_1) \quad (2.22)$$

dengan $k(T_m) = \frac{375}{d_o} \cdot e^{\left(\frac{-556000}{R \cdot T_m}\right)}$.

Kemudian, fraksi gagal ϕ_2 sedemikian rupa sehingga nilainya ≤ 1 . Karena itu, variabel ϕ_2 selanjutnya didefinisikan sebagai persamaan (2.23).

$$\phi_2(t, T) = 1 - e^{-\alpha \cdot \zeta^\beta} \quad (2.23)$$

Nilai α dan β kemudian ditentukan secara empiris. Dan berdasarkan penelitian empiris sebelumnya terhadap partikel UO_2 , diperoleh nilai $\alpha = \ln 2 = 0.693$, sedangkan nilai $\beta = 0.88$.

Dalam TRIAC, faktor fraksi gagal ini tidak akan dipertimbangkan. Hal ini disebabkan karena kondisi ini terjadi pada temperatur di atas 2000°C . Sementara RDE tidak dirancang untuk sampai pada temperatur tersebut.

BAB 3

Penerapan

3.1 Pendahuluan

TRIA *Code* yang telah dijelaskan sebelumnya secara umum dapat dikelompokkan menjadi dua tugas utama, masing-masing adalah perhitungan di waktu irradiasi dan kecelakaan. Penerapannya adalah seperti pada Listing 3.1.

Listing 3.1: triac.py

```
1 import math, sys
2
3 def readdata(namafile):
4     f=open(namafile,"r")
5     statusGeometry=" [m]"
6     statusCharacteristics="SiC Tensile Strength [Pa]"
7     statusIrradiation="INPUT: Irradiation Temp. Hystory"
8     statusAccident="INPUT: Accident Temp. Hystory"
9     statusAll=0
10    dimensi=[]
11    characteristics=[]
12    irradiation=[]
13    accident=[]
14    i=0
15    x=0
16    for baris in f.readlines():
17        i=i+1
18        element=baris.split('\t')
19        if statusAll==0:
20            if element[0]==statusGeometry:
21                for j in range(1,6):
22                    y=float(element[j])
23                    dimensi.append(y)
24                    statusAll=1
25
26        elif statusAll==1:
27            if element[0]==statusCharacteristics:
28                x=i+1
29            elif i==x:
30                try:
31                    for j in range(0,5):
32                        y=float(element[j])
33                        characteristics.append(y)
34                        statusAll=2
35                except:
36                    x=i+1
37
38
39
40        elif statusAll==2:
41            if element[0]==statusIrradiation:
42                x=i+1
```

```

43         elif element[0]==statusAccident:
44             statusAll=3
45         else:
46             temp=[]
47             try:
48                 y=float(element[1])*24*3600
49                 temp.append(y)
50                 y=float(element[2])
51                 temp.append(y)
52                 irradiation.append(temp)
53             except:
54                 x=i+1
55
56
57
58         elif statusAll==3:
59             temp=[]
60             try:
61                 y=float(element[1])*24*3600
62                 temp.append(y)
63                 y=float(element[2])
64                 temp.append(y)
65                 accident.append(temp)
66             except:
67                 x=i+1
68
69     return dimensi, characteristics, irradiation, accident
70
71
72 def OPF(irradiation, dt):
73     x=len(irradiation)
74     y=(17.0/dt)*24*3600
75     tb=1020*24*3600
76     z=0.0
77     for i in range(x):
78         j=irradiation[i]
79         a1=8.3143*j[1]
80         a=-163000/(a1)
81         b=math.exp(a)
82         g=2*(8.32e-11)*b
83         g1=g*(tb-j[0])*y
84         z=z+g1
85
86     Tb=0.85e4/((2*math.log10(tb))-(math.log10(z))-10.08)
87     logDS=-2.3-(8116/Tb)
88     ds=math.pow(10, logDS)
89     tau=ds*tb
90     return z, Tb, ds, tau
91
92 def FTau(tau):
93     looping=0.0
94     for n in range(1, 2000):
95         pangkat=math.pow(n, 2)*math.pow(math.pi, 2)*tau
96         A=math.exp(-(pangkat))
97         B=math.pow(n, 4)*math.pow(math.pi, 4)
98         looping=looping+((1-A)/B)
99
100     ftau=1-((6/tau)*looping)
101     return ftau
102
103 def OPFAccident(Tb, tb, T):
104     logOPF=-10.08-(8500/Tb)+(2*math.log10(tb))-(0.404*((10000/T)-(10000/(Tb+75))))
105     opfa=math.pow(10, logOPF)
106     return opfa
107
108 def Pressure(Tb, dsAccident, Vk, Vf, accident):
109     p=[]
110     for i in range(len(accident)):
111         x=accident[i]
112         y=dsAccident[i]

```

```

113         p.append((y[2]*Ff*OPFAccident(Tb,x[1])*Fb)/((Vf/Vk)*R*x[1]/Vm))
114     return p
115
116 def DS(T):
117     logDS=-2.3-(8116/T)
118     ds=math.pow(10,logDS)
119     return ds
120
121 def interpolasi(a,b,c):
122     x1=a[0]
123     y1=a[1]
124     x2=b[0]
125     y2=b[1]
126
127     i=[]
128     selisih=(x2-x1)/c
129     x=x1
130
131     for k in range(1,c):
132         j=[]
133         x=x+selisih
134         y=((x-x1)/(x2-x1))*(y2-y1)+y1
135         j.append(x)
136         j.append(y)
137         i.append(j)
138     return i
139
140 if __name__=="__main__":
141     if len(sys.argv)==3:
142         f=sys.argv[1]
143         dt=int(sys.argv[2])
144     else:
145         f="example.pan.in"
146         dt=19
147
148     x=readdata(f)
149     dimensi=x[0]
150     characteristics=x[1]
151     irradiation=x[2]
152     accident=x[3]
153
154     VolRef1=(4/3)*math.pi*dimensi[0]*dimensi[0]*dimensi[0]
155     VolRef2=(4/3)*math.pi*dimensi[1]*dimensi[1]*dimensi[1]
156     VolOPyC=VolRef1-VolRef2
157
158     """Volume SiC"""
159     VolRef3=(4/3)*math.pi*dimensi[2]*dimensi[2]*dimensi[2]
160     VolSiC=VolRef2-VolRef3
161
162     """Volume IPyC"""
163     VolRef4=(4/3)*math.pi*dimensi[3]*dimensi[3]*dimensi[3]
164     VolIPyC=VolRef3-VolRef4
165
166     """Volume Buffer & Volume Kernel"""
167     VolKernel=(4/3)*math.pi*dimensi[4]*dimensi[4]*dimensi[4]
168     VolBuff=VolRef4-VolKernel
169     print("Volume OPyC:",VolOPyC)
170     print("Volume SiC:",VolSiC)
171     print("Volume IPyC:",VolIPyC)
172     print("Volume Buffer:",VolBuff)
173     print("Volume Kernel:",VolKernel)
174
175     irradiation2=[]
176     accident2=[]
177     lenIR=len(irradiation)
178     lenAcc=len(accident)
179
180     fi=file('irradiasi2','w')
181     irradiation2.append(irradiation[0])
182     b=0

```



```

183     fi.write(str(b)+' ', '+str(irradiation2[0])+'\n')
184     b=b+1
185     for x in range(1,lenIR):
186         temp=[]
187
188         i=irradiation[x-1][1]
189         j=irradiation[x][1]
190         if i!=j:
191             temp=interpolasi(irradiation[x-1],irradiation[x],dt)
192             for y in range(len(temp)):
193                 irradiation2.append(temp[y])
194                 fi.write(str(b)+' ', '+str(irradiation2[b])+'\n')
195                 b=b+1
196             irradiation2.append(irradiation[x])
197             fi.write(str(b)+' ', '+str(irradiation2[b])+'\n')
198             b=b+1
199
200     fi.close()
201
202     fi=file('accident2','w')
203     b=0
204     fi.write(str(b)+' ', '+str(irradiation2[0])+'\n')
205     b=b+1
206     accident2.append(accident[0])
207     for x in range(1,lenAcc):
208         temp=[]
209
210         i=accident[x-1][1]
211         j=accident[x][1]
212         if i!=j:
213             temp=interpolasi(accident[x-1],accident[x],dt)
214             for y in range(len(temp)):
215                 accident2.append(temp[y])
216                 fi.write(str(b)+' ', '+str(irradiation2[b])+'\n')
217                 b=b+1
218             accident2.append(accident[x])
219             fi.write(str(b)+' ', '+str(irradiation2[b])+'\n')
220             b=b+1
221
222
223     TB=OPF(irradiation2,1)
224     z=TB[0]
225     Tb=TB[1]
226     dsi=TB[2]
227     tauI=TB[3]
228     print("OPF="+str(z)+", Tb="+str(Tb)+", DS="+str(dsi)+", TauI="+str(tauI))
229
230     tb=1020*24*3600
231     Vk=VolKernel
232     Vf=0.5*VolBuff
233     Ff=0.31
234     R=8.3143
235     Vm=2.43796e-5
236     Fb=0.08
237
238     a=(0.5*(pow(dimensi[1],3)+pow(dimensi[2],3)))
239     r=pow(a,(1.0/3))
240     do=dimensi[1]-dimensi[2]
241
242     sigma0=756.e6
243     m=6.93
244     print(FTau(0.05),FTau(0.5),FTau(1),FTau(2))
245     print('ln(2)='+str(math.log(2)))
246     pressure=[]
247     SigmaT=[]
248     phi1=0
249
250     for i in range(len(accident2)):
251         dsa=DS(accident2[i][1])
252         tauA=dsa*accident2[i][0]

```

```

253         if accident2[i][0]==0:
254             Fd=FTau( tauI )
255         else :
256             Fd=(( ( tauI+tauA)*FTau( tauI+tauA ))-(tauA*FTau( tauA )))/ tauI
257
258         opfa=OPFAccident(Tb, tb , accident2[i][1])
259         n=(( Fd*Ff)+opfa)*Fb*(Vk/Vm)
260         p=n*R*accident2[i][1]/Vf
261         pressure.append(p)
262
263         vdot=(5.87e-7)*math.exp(-179500/(8.3143*accident2[i][1]))
264         a=(1+(( vdot*accident2[i][0])/do))
265         sigmaT=(( r*p)/(2*do))*a
266         SigmaT.append(sigmaT)
267
268         a1=sigmaT/sigma0
269         a=pow(a1,m)
270         b=math.exp(-math.log(2)*a)
271         phi=1-b
272         phil=phil+phi

```

3.2 Saat irradiasi

Untuk perhitungan pertama, kita harus bisa mendapatkan nilai OPF (persamaan 2.1), T_B (persamaan 2.4), DS persamaan 2.5) dan τ_i (persamaan 2.6). Pada Listing 3.1, tugas tersebut didelegasikan pada fungsi OPF (irradiation, dt) (baris ke-71 s/d 9). Fungsi tersebut membutuhkan dua argumen, dengan argumen pertama adalah sejarah irradiasi dalam bentuk array. Jika melihat contoh yang disajikan pada Lampiran 3.2, data tersebut terletak setelah baris berisi INPUT: Irradiation Temp. Hystory. Array akan berdimensi dua, yaitu setiap elemen array merupakan array dengan dua elemen, masing-masing adalah waktu dan temperatur irradiasi.

Argumen kedua sejatinya menunjukkan selisih waktu antara pengambilan data yang satu dengan pengambilan data berikutnya. Nilai tersebut diterapkan dalam Listing 3.1 sebagai variabel y. Karena selisih waktu antar pengambilan data yang dimaksud pada contoh Lampiran 3.2 dinilai cukup tinggi (17 hari), maka interpolasi dicoba untuk diterapkan juga. Hal ini dilatarbelakangi hasil simulasi yang disajikan dalam literatur [4]. Sebagai ilustrasi, jika tanpa interpolasi, nilai temperatur irradiasi $T_B = 1057.41^\circ\text{C}$. Bila dijalankan dengan menggunakan skenario selisih antar pengambilan data adalah $\frac{17}{2} = 8.5$ hari, dihasilkan nilai $T_B = 1046.55^\circ\text{C}$. Sedangkan, nilai T_B yang dihitung dengan skenario selisih antar pengambilan data adalah 1 hari (interpolasi jumlah pembagian sebanyak 17 menghasilkan nilai 1040.07°C . Nilai T_B selanjutnya akan semakin kecil dengan semakin kecilnya jarak pengambilan data (semakin besar pembagian untuk interpolasi).

Selanjutnya

Daftar Referensi

- [1] J. Wang, “An integrated performance model for high temperature gas cooled reactor coated particle fuel,” Ph.D. dissertation, Massachusetts Institute of Technology, 2004.
- [2] “Reaktor daya eksperimental (rde),” <http://www.batan.go.id/index.php/id/reaktor-daya-eksperimental-rde>, diakses: 17-07-2017.
- [3] K. Verfondern, J. Cao, T. Liu, and H.-J. Allelein, “Conclusions from v&v studies on the german codes panama and fresco for htgr fuel performance and fission product release,” *Nuclear Engineering and Design*, vol. 271, pp. 84 – 91, 2014, sI : HTR 2012. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0029549313005992>
- [4] K. Verfondern and H. Nabielek, “The mathematical basis of the panama-i code for modelling pressure vessel failure of triso coated particles under accident conditions,” Julich Research Center, Germany, Tech. Rep., 1990.

LAMPIRAN

Lampiran 1

TRIAC-BATAN

TRISO Analysis Code of BATAN

"Developed by Computational Laboratory, Center for Nuclear Reactor Technology and Safety, BATAN"

Case Title: (describe your problem case here)

TRISO Geometry:

Outer radius CFP SiC IPyC buffer kernel center

[m] 4.60E-04 4.20E-04 3.85E-04 3.45E-04 2.50E-04 0

Properties and Operation Parameters:

SiC Tensile Strength [Pa] Weibull Modulus Burnup [FIMA] "Fission Yield
of stable fission gasses, Ff" Fast Neutron Fluence Weight ratio of th to U-
235 in kernel

8.34E+08 8.02 0.09 0.31 2.4

Properties and Operation Parameters related with thermal decomposition:

Alpha Beta

0.0001 4

-1 1401.6 0.1 10

INPUT: Irradiation Temp. Hystory

1	0	593
2	17	833
3	34	1023
4	51	1093
5	68	1123
6	85	833
7	102	1023
8	119	1093
9	136	1123
10	153	833
11	170	1023
12	187	1093
13	204	1123
14	221	833
15	238	1023
16	255	1093
17	272	1123
18	289	833
19	306	1023
20	323	1093
21	340	1123
22	357	833
23	374	1023
24	391	1093
25	408	1123
26	425	833
27	442	1023
28	459	1093
29	476	1123
30	493	833
31	510	1023
32	527	1093
33	544	1123
34	561	833
35	578	1023
36	595	1093
37	612	1123

38	629	833
39	646	1023
40	663	1093
41	680	1123
42	697	833
43	714	1023
44	731	1093
45	748	1123
46	765	833
47	782	1023
48	799	1093
49	816	1123
50	833	833
51	850	1023
52	867	1093
53	884	1123
54	901	833
55	918	1023
56	935	1093
57	952	1123
58	969	833
59	986	1023
60	1003	1093
61	1020	1123

0

-1 180 1

INPUT: Accident Temp. Hystory

1	0	1033
2	0.0271	1033
3	0.2208	1068
4	1	1160
5	10	1571
6	20	1728
7	30	1752
8	35	1749
9	60	1690
10	90	1605
11	120	1526
12	180	1395

0