



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>	8
2.3 Menghitung OPF saat irradiasi	9
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
2.2	Contoh sejarah irradiasi	9

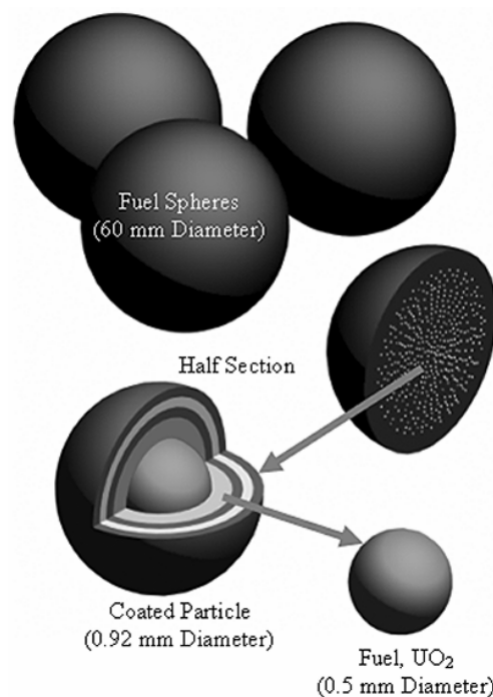
Daftar Program

2.1	triac.py	5
-----	--------------------	---

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.

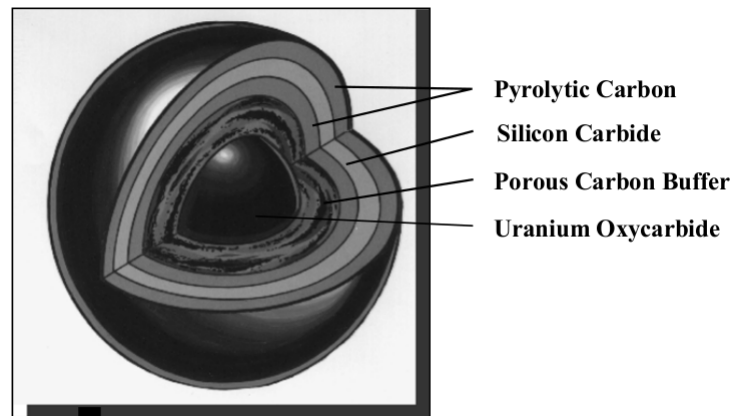


Gambar 1.1: Ilustrasi bentuk bahan bakar *pebble* [1]

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*).



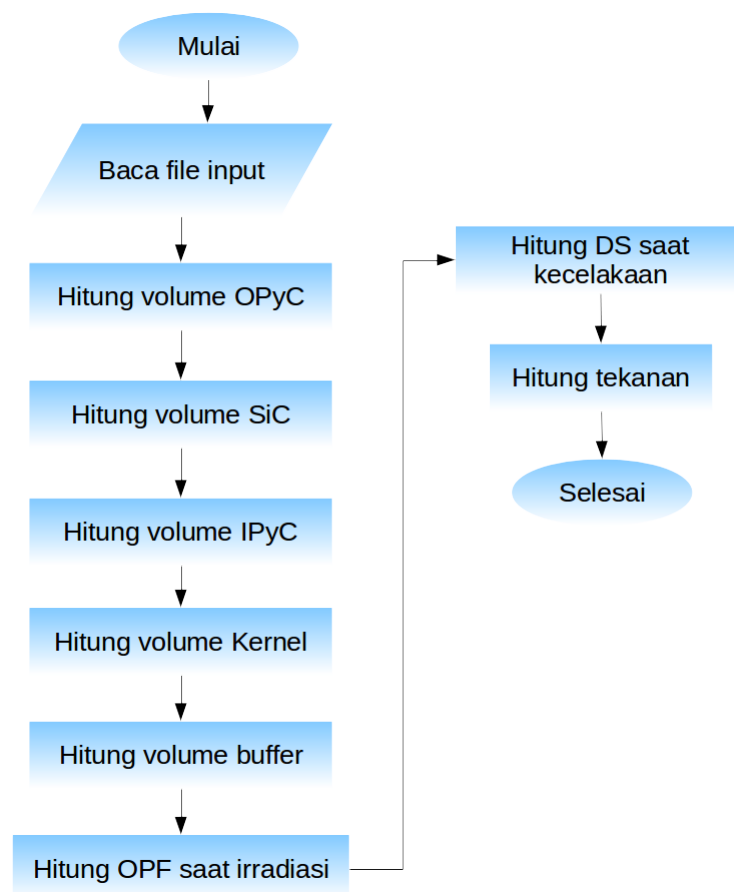
Gambar 1.2: Komposisi elemen pelapis partikel [1]

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 2.1.



Gambar 2.1: Diagram alir perhitungan TRIAC

Listing 2.1: 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             elif statusAll==2:
39                 if element[0]==statusIrradiation:
40                     x=i+1
41                 elif element[0]==statusAccident:
42                     statusAll=3
43             else:
44                 temp=[]
45                 try:
46                     y=int(element[0])
47                     temp.append(y)
48                     y=float(element[1])
49                     temp.append(y)
50                     y=int(element[2])
51                     temp.append(y)
52                     irradiation.append(temp)
53                 except:
54                     x=i+1
55
56             elif statusAll==3:
57                 temp=[]
58                 try:
59                     y=int(element[0])
60                     temp.append(y)
61                     y=float(element[1])
62                     temp.append(y)
63                     y=int(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     print("Irradiation length:",x)
75     y=dt*24*3600
76     z=0.0
77     for i in range(1,x-1):
78         j=irradiation[i]
79         g=2*(8.32e-11)*(math.exp(-163000/(8.3143*j[2])))*(1020-j[1])*24*3600*y
80         z=z+g
81
82     """ print("OPF: ", z) """
83
84     tb=1020*24*3600
85     Tb=0.85e4/((2*math.log10(tb))-(math.log10(z))-10.08)
86     """ print("Tb: ", Tb) """
87     logDS=-2.3-(8116/Tb)
88     ds=math.pow(10,logDS)
89     """ print("DS: ", ds) """
90     tauI=ds*tb
91     """ print("tauI: ", tauI) """
92     return z,Tb,ds,tauI
93
94 def FD(tauA ,tauI):
95     summasi=tauA+tauI
96     looping=0.0
97     loopingPlus=0.0
98     loopingI=0.0
99     for n in range(1,2000):
100         pangkat=math.pow(n,2)*math.pow(math.pi,2)*tauA
101         A=math.exp(-(pangkat))
102         B=math.pow(n,4)*math.pow(math.pi,4)
103         looping=looping+((1-(A)/(B)))
104
105         pangkatI=math.pow(n,2)*math.pow(math.pi,2)*tauI
106         AI=math.exp(-(pangkatI))
107         loopingI=loopingI*((1-(AI)/(B)))
108         if tauA==0:
109             """ print(n, pangkatI, AI, B, AI/B, loopingI) """
110             loopingPlus=loopingPlus+(1-(math.exp(-(math.pow(n,2)*math.pow(math.pi,2)*summasi)))/(
111         if tauA==0:
112             FTau=1-((6/tauI)*loopingI)
113             print('tauI=',tauI)
114             print('loopingI=',loopingI)
115             print('ftauI=',FTau)
116             Fd=FTau
117         else:
118             FTau=1-((6/tauA)*looping)
119             FTauPlus=1-((6/summasi)*loopingPlus)
120             Fd=((summasi*FTauPlus)-(tauA*FTau))/tauI
121     return Fd
122
123 def OPFAccident(Tb,T):
124     tb=1020*24*3600
125     logOPF=-10.08-(8500/Tb)+(2*math.log10(tb))-(0.404*((10000/T)-(10000/(Tb+75))))
126     return math.pow(10,logOPF)
127
128 def Pressure(Tb,dsAccident ,Vk,Vf,accident):
129     Ff=0.31
130     R=8.3143
131     Vm=2.43796e-5
132     Fb=0.08
133     p=[]
134     for i in range(len(accident)):
135         x=accident[i]
136         y=dsAccident[i]
137         p.append((y[2]*Ff*OPFAccident(Tb,x[2])*Fb)/((Vf/Vk)*R*x[2]/Vm))
138     return p
139

```

```

140 def DS(accident , tauI ):
141     x=len( accident )
142     ds=[]
143     for i in range(x):
144         j=accident[ i ]
145         logDS=-2.3-(8116/j [2])
146         k=[]
147         l=math.pow(10 , logDS )
148         k.append(1)
149         tauA=l*j [1]*24*3600
150         k.append( tauA )
151         Fd=FD( tauA , tauI )
152         k.append( Fd )
153         ds.append(k)
154     return ds
155
156 if __name__=="__main__":
157     if len( sys . argv )==3:
158         f=sys . argv [1]
159         dt=int( sys . argv [2])
160     else :
161         f="example . pan . in "
162         dt=17
163
164     x=readdata( f )
165     dimensi=x[0]
166     characteristics=x[1]
167     irradiation=x[2]
168     accident=x[3]
169     """ print( "Dimensi:" , dimensi )
170     print( " Karakteristik:" , characteristics )
171     print( " Irradiasi:" , irradiation )
172     print( " Accident:" , accident ) """
173     """Volume Outer Pyrolytic Carbon """
174     VolRef1=(4/3)*math.pi*dimensi[0]*dimensi[0]*dimensi[0]
175     VolRef2=(4/3)*math.pi*dimensi[1]*dimensi[1]*dimensi[1]
176     VolOPyC=VolRef1-VolRef2
177
178     """Volume SiC """
179     VolRef3=(4/3)*math.pi*dimensi[2]*dimensi[2]*dimensi[2]
180     VolSiC=VolRef2-VolRef3
181
182     """Volume IPyC """
183     VolRef4=(4/3)*math.pi*dimensi[3]*dimensi[3]*dimensi[3]
184     VolIPyC=VolRef3-VolRef4
185
186     """Volume Buffer & Volume Kernel """
187     VolKernel=(4/3)*math.pi*dimensi[4]*dimensi[4]*dimensi[4]
188     VolBuff=VolRef4-VolKernel
189     print( "Volume OPyC:" , VolOPyC)
190     print( "Volume SiC:" , VolSiC)
191     print( "Volume IPyC:" , VolIPyC)
192     print( "Volume Buffer:" , VolBuff)
193     print( "Volume Kernel:" , VolKernel)
194     TB=OPF( irradiation , dt)
195     z=TB[0]
196     Tb=TB[1]
197     ds=TB[2]
198     tauI=TB[3]
199     print( "OPF=" , z , "Tb=" , Tb , "DS=" , ds , "TauI=" , tauI )
200     dsAccident=DS( accident , TB[3])
201     for i in range(len( dsAccident )):
202         x=dsAccident[ i ]
203         print(x[0] , x[1] , x[2])
204
205     p=Pressure( Tb , dsAccident , VolKernel , VolBuff/2 , accident )
206     print(p)

```

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 2.1

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 2.1). 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 2.1).
2. Data tentang kekuatan SiC. Data ini diidentifikasi menggunakan teks yang didefinisikan oleh variabel `statusCharacteristics` (baris ke-6 pada Listing 2.1). 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 2.1).
3. Data tentang sejarah iradiasi. Data ini diidentifikasi menggunakan teks yang didefinisikan oleh variabel `statusIrradiation` (baris ke-7 pada Listing 2.1). 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 tiga elemen yang mewakili data tiap kolom pada setiap akuisisi (baris ke-12 pada Listing 2.1). Ilustrasinya adalah seperti `[[1,0,593],[2,17,833],...]`
4. Data tentang sejarah kecelakaan yang dalam hal ini adalah kondisi di mana temperatur bahan bakar lebih besar daripada 2000°C. Data ini diidentifikasi menggunakan teks yang didefinisikan oleh variabel `statusAccident` (baris ke-8 pada Listing 2.1). 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 2.1). Ilustrasinya adalah seperti `[[1,0,1033],[2,0.0271,1033],...]`

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 7 hari, maka Δ_i adalah 17 hari atau $17 \times 24 \times 3600$ detik.

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

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 konstantan gas sebesar $8.3143 \left[\frac{J}{mole \cdot K} \right]$.

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

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

Sebagai contoh, sejarah irradiasi untuk 10 pencatatan pertama disajikan Gambar 2.2. Dari Gambar 2.2, pada pencatatan pertama, yaitu hari ke-0, temperatur bahan bakar adalah $593^\circ C$. Selanjutnya, pada hari ke-17, temperatur bahan bakar adalah $833^\circ C$, dst.

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

Gambar 2.2: Contoh sejarah irradiasi

Untuk menghitung nilai OPF

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>

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