12-2002

REACLIBaLIVe! (REACLIB Rate Library Interactive Viewer): A Software Package for Graphical Analysis of Nuclear Reaction Rates for Astrophysics

Eric J. Lingerfelt
University of Tennessee - Knoxville

To the Graduate Council:

I am submitting herewith a thesis written by Eric J. Lingerfelt entitled "REACLIBaLIVe! (REACLIB Rate Library Interactive Viewer): A Software Package for Graphical Analysis of Nuclear Reaction Rates for Astrophysics." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Physics.

Mike Guidry, Major Professor

We have read this thesis and recommend its acceptance:

Kermit Duckett, William Raphael Hix, Michael Smith, Marianne Breinig

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

The Graduate Counsel:

I am submitting herewith a thesis written by Eric J. Lingerfelt entitled "REACLIB aLIVe! (REACLIB Rate Library Interactive Viewer): A Software Package for Graphical Analysis of Nuclear Reaction Rates for Astrophysics." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Physics.

Mike Guidry____________________

Major Professor

We have read this thesis and recommend its acceptance:

Kermit Duckett__________

William Raphael Hix_____

Michael Smith__________

Marianne Breinig________

Accepted for the Council:

Anne Mayhew_________

Vice Provost and Dean of Graduate Studies

(Original signatures are on file with official student records)

REACLIB aLIVe!
(REACLIB Rate Library Interactive Viewer): A Software Package for Graphical Analysis of Nuclear Reaction Rates for Astrophysics

A Thesis Presented for the
Master of Science Degree
The University of Tennessee, Knoxville

Eric J. Lingerfelt

December 2002

Dedication

This thesis is dedicated to my grandfather, Samuel Lingerfelt, whose spark I bear.

“All energy flows to the whim of the great magnet” - Dr. H. S. Thompson

Acknowledgements

I would like to thank my advisors Dr. Mike Guidry, Dr. William Raphael Hix, and Dr. Michael Smith for the insight, guidance, and encouragement I received. In addition I would like to thank the other members of my committee: Dr. Kermit Duckett and Dr. Marianne Breinig for their help and guidance with the development of my thesis.

Special thanks to members of my group and the department: Erin McMahon, Suzanne Perete-Koon, Murat Ozer, Robert Mahurin, Luc Dessieux, Dr. Wayne Kincaid, James Wicker, Khaled Mriziq, and Rodney Sullivan for their encouragement and help during the development of this thesis.

I would also like to thank Dr. Lee Riedinger and Dr. Soren Sorenson, the former and present head of the physics and astronomy department, for the award of teaching assistantship and the Science Alliance Fellowship, while I pursued this degree and also Dr. Mike Guidry, Dr. Marianne Breinig, and the Tennessee Educational Technology Initiative for the award of research assistantship.

I would like to thank my dad, David G. Lingerfelt, and my mom and her husband, Eula K. Keplinger and Ronald Keplinger for their spiritual (and financial) support in pursuit of this degree.

Thanks to: Travis Miller, “Dangerous” Daniel Dugger, Jonathan Kegley, Allison Hollier, Norris Guthrie, Jr., Jason Watts, Katherine Poland, Manya Whitney, and Dr. Velvet “Spicy” Jones for their continuous encouragement and support I received while pursuing this degree.

Most of all I would like to thank my better half Sarah E. Page without whom none of this would be possible.

Abstract

Nucleosynthesis occurs in such diverse astrophysical phenomena as ordinary stars, like our own Sun, supernovae, novae, X-ray bursts, and the Big Bang. Large sets of nuclear reaction rates for hundreds of seed isotopes are utilized in simulations of these nucleosynthesis processes. A cross-platform, Java software package called REACLIB aLIVe! has been developed with intuitive graphical interfaces and interactive controls to produce custom one-dimensional plots of reaction rates. The points used for these plots are calculated from exponential fits whose parameters, along with other quantities, make up the REACLIB Nuclear Reaction Rate Library. The software offers nuclear astrophysicists the capability to rapidly display any of 8000 nuclear reactions in the library, as well as to add new reaction rates and compare them to ones in the library. The plots produced by the software may be exported in the postscript format, which is easily edited and incorporated into papers, presentations, and websites. The software is available over the World Wide Web or as a downloadable Java archive file.

Contents

List of Figures

Chapter 1

Introduction

1.1 Thesis Foreword

One of the fundamental problems in astrophysics is determining the process by which chemical elements are produced. In 1948, George Gamow and his collaborators proposed the hot Big Bang Theory. However, the instability of nuclei with atomic weights of 5 to 8 limited the Big Bang to producing hydrogen, helium and some lithium, boron, and beryllium. In the 1950's and 1960's, the predominant theory regarding the formation of the chemical elements in the Universe was formulated, largely due to the work of Burbridge, Burbridge, Fowler, and Hoyle [1]. The BBFH theory, as it came to be known, postulated that all elements not produced in the Big Bang or cosmic rays were produced in stars.

The nucleosynthesis that takes place inside normal stars is responsible for the heavier elements that we see today. The lowest mass stars can only synthesize helium. Stars around the mass of our Sun can synthesize helium, carbon, and oxygen. Massive stars (M>5 solar masses) can synthesize elements up to iron. Approximately one-half of the isotopes heavier than iron are produced in red giant stars by the slow neutron capture process.

Type II supernovae result from massive supergiant stars that have quickly evolved to the stage where many concentric layers of nuclear fusion are found in the interior of the star. When the iron core of a super giant star collapses due to gravity and rebounds, an energetic wave of neutrinos is produced, and the outer shells explode into space. In this setting, it is thought, are nearly half of the heavy neutron-rich isotopes that we see in our solar system and the Universe formed.

Collaborative scientific endeavors around the world have studied the problem of element synthesis. In calculating the thousands of nuclear reactions that must take place to simulate astrophysical phenomena, large libraries of experimental and theoretical nuclear reaction rates and parameters are utilized. The most widely used library is REACLIB, containing 8000 reaction rates [2]. The library, which is updated periodically with new or refined rates, is enormous and cannot be examined closely without great inconvenience.

Current technologies exist to rapidly develop computer programs that can assist nucleosynthesis research. With modern tools one can write, debug, and distribute a software package locally or around the globe via the World Wide Web in a relatively small amount of time. We have created such software to greatly improve the utility of REACLIB.

The software, REACLIB aLIVe! (Library Interactive Viewer), is constructed of cross-platform, portable, extendable, and reusable modules written in the Java programming language. Through advanced, intuitive graphical user interfaces, aLIVe! allows the nuclear scientist to quickly explore and graphically analyze the content within REACLIB. The user also has the capability to add new reaction rates and compare them to ones in the library. The plots produced by the software may be exported in the postscript format, which is easily edited and incorporated into papers, presentations, and websites.

The new technology presented in this thesis promises to assist nucleosynthesis calculations by offering new insight into competing reaction rates, expediting changes to the library, and providing an educational tool for those studying nucleosynthesis. In this thesis we will examine nucleosynthesis, the astrophysical events responsible for element production, the user interfaces in aLIVe!, and computer programming issues and techniques applied in the construction of the software.

1.2 Preliminary Concepts

1.2.1 Notation and Definitions

In an interaction between a charged particle, α , and its target, X , a state called the compound nucleus may momentarily exist. A compound nucleus, represented by $Z*$ in the equation

$$\alpha + X \rightarrow Z* \rightarrow Y + \beta ,$$

is nearly always highly excited and quickly splits into possible products. In the example shown here, we assume two such products β and Y . The equation can also be represented by $X(\alpha,\beta)Y$. This reaction could have resulted in several different product combinations, or *exit channels*. By Heisenberg's uncertainty principle, each exit channel, i , has a corresponding lifetime for decay, τ_i , and an associated energy interval, or *width* Γ_i , such that

$$\Gamma_i \tau_i = \eta .$$

The probability, P_i, that exit channel i will occur from j possibilities is

$$P_i = \frac{1/\tau_i}{\sum\limits_{j \neq i} 1/\tau_j} = \frac{\tau}{\tau_i}$$

or

$$P_i = \frac{\Gamma_i}{\Gamma}.$$

Here τ is the *total mean lifetime*, while Γ is the *total energy width*.

1.2.2 Nuclear Energy

The amount of energy required to break up a nucleus into its constituent neutrons and protons and move them apart by an infinite distance is called the *binding energy*. The binding energy, B, of an atomic nucleus can be written as

$$B = (\textit{total mass of indiviual nucleons - total mass of the atomic nucleus})c^2 .$$

The curve of binding energy, where the amount of binding energy per nucleon (B/A) is plotted versus the atomic mass number A, is shown in Figure 1.1 [3]. This curve indicates the amount of stability of all atomic nuclei [4]. At a peak near A = 60 lie the most stable nuclei, the *iron peak* nuclei.

The curve suggests that the heaviest nuclei are less stable than the iron peak nuclei, which indicates that energy can be released if these nuclei are split apart. The heaviest elements are then fissionable. The nuclei of the lightest elements are also less stable than the iron peak nuclei. Thus, fusing two light nuclei like hydrogen and helium into a heavier nucleus can also produce energy.

The energy released in fission and fusion reactions is called the *Q value*. For a fusion reaction $\alpha + X \rightarrow Y$, we can write

$$B_\alpha + B_X = B_Y + Q$$

and for a fission reaction $X \rightarrow Y + \beta$

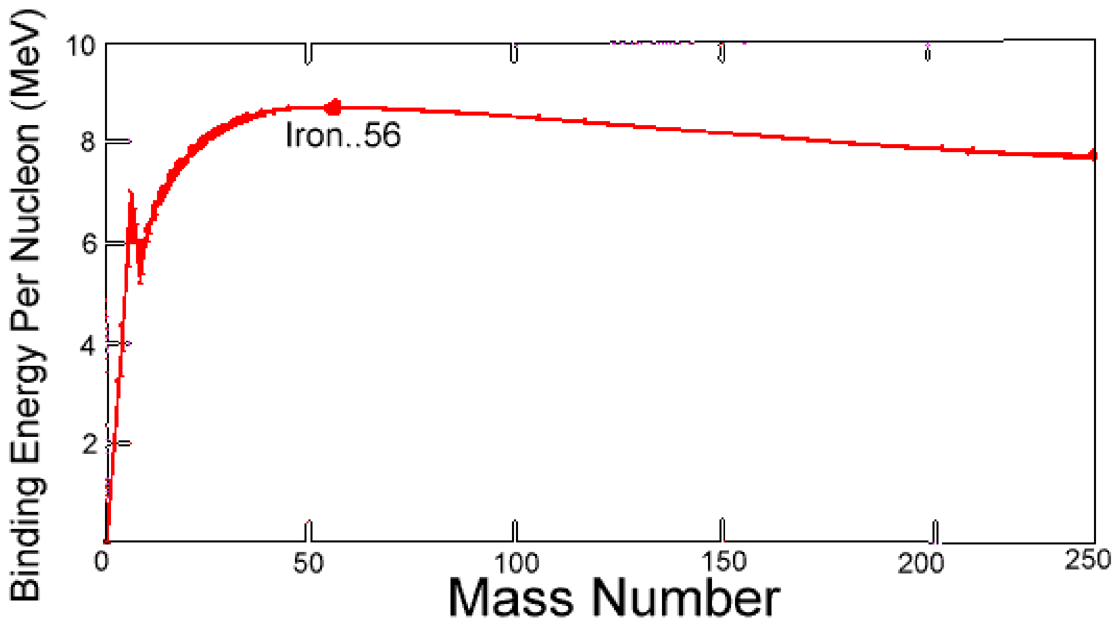


Figure 1.1 The curve of binding energy

$$B_X = B_Y + B_\beta + Q\,.$$

In order for a charged-particle interaction to take place, the colliding particles must overcome the *Coulomb barrier*. By defining the barrier penetration factor P_l as the probability that a particle may quantum mechanically tunnel through the Coulomb barrier, we have a relationship between Γ_i and a factor γ_i^2 such that for charged particles

$$\Gamma_i = 2P_l\gamma_i^2$$

where γ_i^2 is known as the *reduced width* and l is the angular momentum quantum number. The reduced width is the probability that the compound nucleus is composed of the charged particles of the *i*th channel in a common nuclear potential.

In astrophysical circumstances, the entrance channel energies are usually much less than the Coulomb barrier height B_C, therefore the barrier penetration factor P_l depends greatly on the channel energy ε. This relationship, which can be found in the literature, is approximated by

$$P_l(\varepsilon) \propto e^{-2\pi\eta}$$

where the dimensionless Sommerfeld factor η is defined as

$$\eta = \frac{Z_\alpha Z_X e^2}{\mathrm{\eta} v} = 0.1574 Z_\alpha Z_X \left(\frac{\mu}{\varepsilon}\right)^{1/2}$$

where μ is the reduced mass, *e* is the elementary charge, and Z_α and Z_X are the proton numbers of the reactants [5].

1.2.3 The Chart of the Nuclides

If we plot all isotopes by their proton and neutron number, we generate the *Chart of the Nuclides*. A portion of a nuclide chart is shown below in Figure 1.2. Here the number in the superscript of each isotope label is the atomic mass A, which is the sum of the number of neutrons N and protons Z in each nucleus.

1.2.4 Vectors in the NZ-Plane

We may characterize nuclear reactions by vectors in the NZ-plane beginning with the reactant nucleus and ending with the product nucleus. By plotting proton number Z versus neutron number N, the nuclear reactions can be represented graphically as in Figure 1.3. Here we see twelve different types of nuclear reactions. They are characterized by the release or capture of protons, neutrons, helium nuclei (alpha particles), and photons (gamma particles) [6].

1.2.5 The Valley of Stability

The valley of stability, as seen in Figure 1.4, is the group of isotopes that are stable against radioactive beta decay [7]. For isotopes with a small atomic mass, a stable nuclear state corresponds to equal numbers of neutrons and protons, but as we travel up the valley isotopes require more neutrons than protons in their nucleus to balance the Coulomb repulsion between protons with the effects of the strong nuclear force.

Unstable isotopes that lie between the neutron and proton drip lines, but not within the valley of stability, may exist for short lifetimes before beta decay occurs. For isotopes beyond the neutron drip line, neutron emission will likely take place due to their weak binding energies. Likewise, isotopes beyond the proton drip line are unstable with respect to proton emission. Since isotopes outside of the region are very unstable, the study of element production in astrophysics is usually limited to isotopes between the drip lines. The location of the drip lines is, however, unknown and the focus of international research efforts.

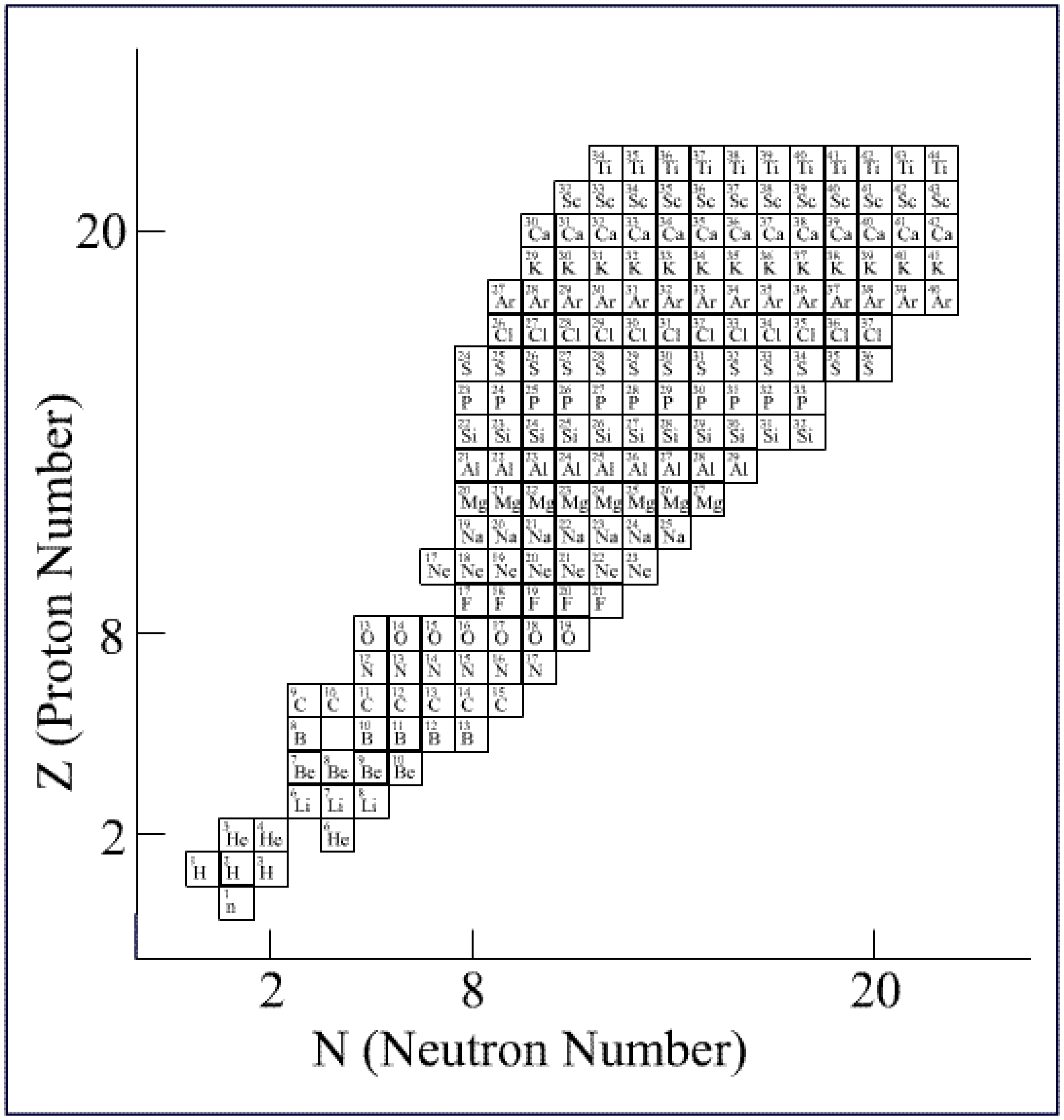


Figure 1.2 A portion of the chart of the nuclides

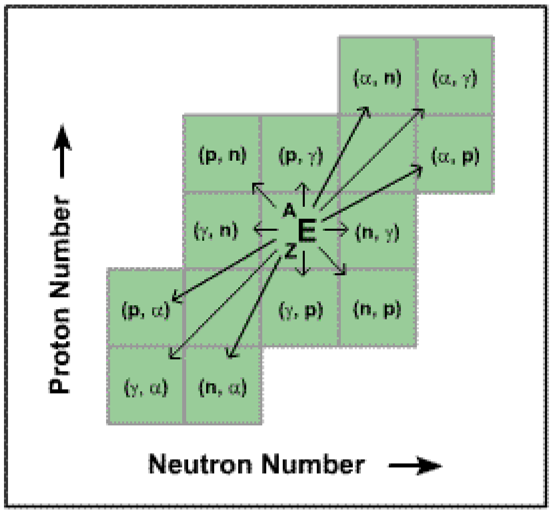


Figure 1.3 Schematic of twelve nuclear reactions represented by vectors in the NZ-plane

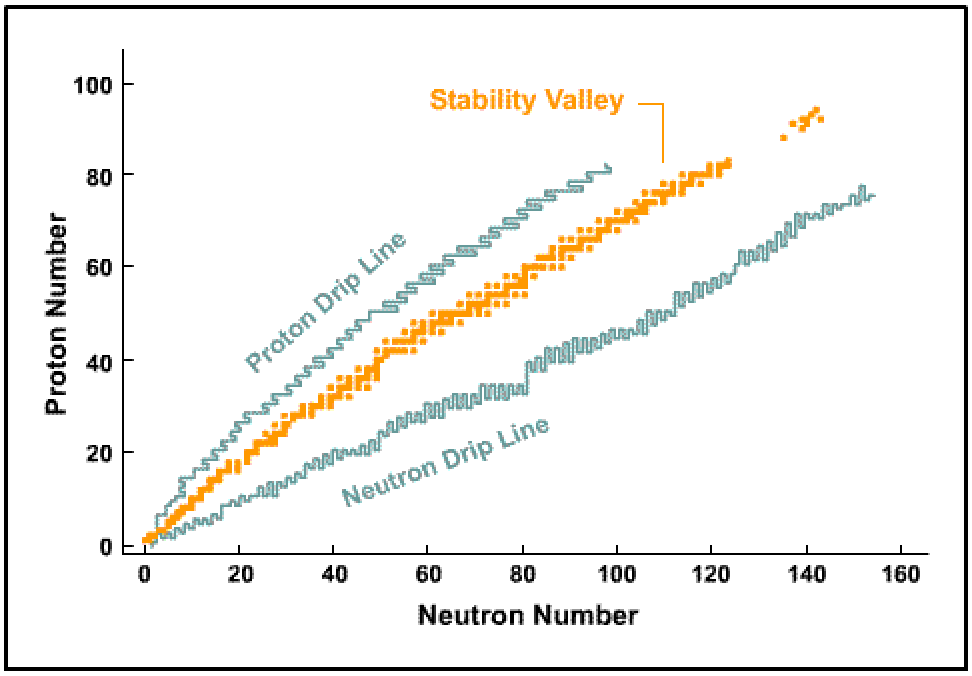


Figure 1.4 The valley of beta stability

1.3 Nucleosynthesis Calculations

1.3.1 Thermonuclear Reaction Rates

The reaction rate for a thermonuclear reaction consisting of the target X and projectiles α with a single velocity v depends partly on the reaction's cross-section $\sigma_{\alpha\beta}$ where β is the outgoing product. The cross section can be written as

$$\sigma_{\alpha\beta}(v) = \frac{\text{number of reactions per unit time per target } X}{\text{incident flux of projectiles } \alpha} \text{cm}^2 .$$

Solving for the number of reactions per unit time we find that

$$r_{\alpha\beta}(v) = n_\alpha n_X \sigma_{\alpha\beta}(v) v \ \text{cm}^{-3}\ \text{s}^{-1}$$

where n_a and n_X are the number density of reactants. If X and α are the same species, a factor of $(1-\delta_{\alpha X})^{-1}$ must be applied, where $\delta_{\alpha X}$ is the Kroenecker delta, to ensure that no double counting will take place. This would ensure that in the collision between two identical particles the rate would be divided by two.

In most astrophysical situations, we will deal with large ensembles of particles having a Maxwell-Boltzmann distribution of velocities or energies. By integrating over the particles in each distribution, we arrive at

$$r_{\alpha\beta}(v) = n_\alpha n_X \langle \sigma \ v \rangle_{\alpha\beta} \text{cm}^{-3}\text{s}^{-1} .$$

Here the product $\langle \sigma v \rangle_{\alpha\beta}$ is weighted by the differential Maxwell-Boltzmann distribution,

$$\langle \sigma v \rangle_{\alpha\beta} \equiv \frac{\int_0^\infty \Psi(\varepsilon)\sigma_{\alpha\beta}\upsilon d\varepsilon}{\int_0^\infty \Psi(\varepsilon)d\varepsilon}$$

with

$$\Psi(\varepsilon) = \frac{2}{\pi^{1/2}}\frac{1}{(kT)^{3/2}} e^{-\varepsilon/kT}\varepsilon^{1/2}$$

where ε is the center of mass energy of the reactants. By setting $v = \left(2\varepsilon/m\right)^{1/2}$, we obtain

$$\langle \sigma v \rangle_{\alpha\beta} = \left(\frac{8}{\pi m}\right)^{1/2} \left(kT\right)^{-3/2} \int_0^\infty \sigma_{\alpha\beta}(\varepsilon)\, e^{-\varepsilon/kT}\ \varepsilon\, d\varepsilon \ \text{cm}^3\ \text{s}^{-1}$$

where the reduced mass m is defined as

$$m = {m_\alpha m_X}/{\left(m_\alpha + m_X\right)}.$$

We must now calculate $\sigma_{\alpha\beta}\left(\varepsilon\right)$ in the center of mass system. The cross section can be written as

$$\sigma_{\alpha\beta}\left(\varepsilon\right) = \pi\Delta^2 g \frac{\Gamma_\alpha \Gamma_\beta}{\Gamma^2} f\left(\varepsilon\right)$$

where Γ_α and Γ_β are the widths of the entrance and exit channels, respectively. The wavelength Δ is the reduced DeBroglie Wavelength

$$\pi\Delta^2 = \frac{\pi\eta^2}{2\varepsilon m} = \frac{0.657}{\varepsilon\,(\text{MeV})}\frac{1}{\mu}\,\text{barns}\,(10^{-24}\ \text{cm}^2)$$

where μ is the reduced mass in amu's and g is a statistical factor that holds information concerning the spins of the reactants, products, and the compound nucleus.

The factor $f(\varepsilon)$, known as the *shape factor*, is either a *non-resonant* or *resonant* form. A resonant shape factor changes quickly over a range of energies, while a non-resonant factor changes slowly with energy [5].

1.3.2 Non-Resonant Reactions

If the shape factor $f(\varepsilon)$ is slowly varying, then we have a non-resonant reaction. This occurs when ε is much different than any resonant energy ε_r. The non-resonant form of the cross section often is

$$\sigma_{\alpha\beta}(\varepsilon) = \frac{S(\varepsilon)}{\varepsilon} e^{-2\pi\eta}$$

where $S(\varepsilon)$ is slowly varying with ε. The $1/\varepsilon$ term is associated with the DeBroglie wavelength and the exponential term is from the penetrability factor P_l [5].

1.3.3 Resonant Reactions

A resonant shape factor may often be expressed in the form of a Breit-Wigner resonance,

$$f(\varepsilon) = \frac{\Gamma^2}{(\varepsilon - \varepsilon_r)^2 + \left(\Gamma/2\right)^2}.$$

A strong peak occurs around the resonance energy ε_r. If, in the interval $\varepsilon_r - \Gamma/2 \le \varepsilon \le \varepsilon_r + \Gamma/2$, the total energy width Γ does not change a great deal, then the width at half maximum is Γ. Hence,

$$\sigma_{\alpha,\beta} = \pi \Delta^2 g \frac{\Gamma_\alpha \Gamma_\beta}{(\varepsilon - \varepsilon_r)^2 + \left(\Gamma/2\right)^2}.$$

The cross section is dominated by the $(\varepsilon - \varepsilon_r)^2 + \left(\Gamma/2\right)^2$ term. If $\varepsilon_r >> \Gamma/2$, the distribution $\Psi(\varepsilon)$ and the total energy width Γ vary little, and so they are evaluated at ε_r [5]. By integrating the denominator over an infinite range of energies, we find an approximation for the thermonuclear product $\langle \sigma v \rangle_{\alpha\beta}$, which can given by

$$\langle \sigma v \rangle_{\alpha\beta} = \eta^2 \left(\frac{2\pi}{mkT} \right)^{3/2} g \frac{\Gamma_\alpha \Gamma_\beta}{\Gamma} e^{-\varepsilon_r / kT}.$$

1.3.4 Thermonuclear Reaction Networks

We may group various types of reactions into three categories based on the number of nuclear reactants. Decays, electron and positron captures, photodisintegrations, and neutrino-induced reactions make up one group, while two and three nuclei reactions produce the other two. The derivative with respect to time of the number densities of each nuclear reactant can be written as a function of the reaction rate *r* as in

$$\left. \frac{\partial n_i}{\partial t} \right|_{\rho = const} = \sum_j \mathcal{N}^i_j r_j + \sum_{j,k} \mathcal{N}^i_{j,k} r_{j,k} + \sum_{j,k,l} \mathcal{N}^i_{j,k,l} r_{j,k,l}$$

where

$$\mathcal{N}^{i}_{j} = N_i ,$$

$$\mathcal{N}^{i}_{j,k} = N_i \Big/ \prod_{m=1}^{n_m} \left|N_{j_m}\right|! ,$$

and $$\mathcal{N}^{i}_{j,k,l} = N_i \Big/ \prod_{m=1}^{n_m} \left|N_{j_m}\right|! \quad .$$

Here the N_i terms indicate the number of particles of type *i* that are created or destroyed. The factorial terms within each sum prevent double counting of identical particle reactions.

The astrophysical plasma in which nuclear reactions take place may expand or contract in volume. This effect also changes the number densities of the interacting species. By introducing the nuclear abundance $Y_i = n_i / \rho N_A$, this volume effect can be separated from the nuclear ones. In this expression, a reactant *i* with an atomic weight of A_i has a mass fraction of $A_i Y_i$. Hence,

$$\sum A_i Y_i = 1 .$$

By rewriting the above formula in terms of the abundances and nuclear cross sections, we arrive at a set of ordinary differential equations for the time rate of change for the abundances Y_i [8]. The reaction network is then written

$$\dot{Y}_i = \sum_j \mathcal{N}^{i}_{j} \lambda_j Y_j + \sum_{j,k} \mathcal{N}^{i}_{j,k} \rho N_A \langle j,k \rangle Y_j Y_k + \sum_{j,k,l} \mathcal{N}^{i}_{j,k,l} \rho^2 N_A^2 \langle j,k,l \rangle Y_j Y_k Y_l .$$

1.4 The REACLIB Nuclear Reaction Rate Library

The REACLIB reaction rate library contains theoretical and experimental reaction rates for 5411 seed isotopes from a variety of sources and is updated periodically [9]. The library is split into eight categories that depend on the number of reactants and products in each reaction. Table 1.1 lists the eight reaction types.

Each of the over 8,000 rates in REACLIB has three lines of information, as seen in the example given in Figure 1.5 [2]. The first line identifies the nuclei involved by element symbol and atomic mass, the source of information for each reaction, and the Q value in MeV. (Table 1.2 lists the abbreviations used for these sources [10].)

The reaction rates are defined by the seven parameters $(a_0$ through $a_6)$ in the next line by

$$\text{Reaction Rate} = \exp\left(a_0 + a_1 T_9^{\,1} + a_2 T_9^{\,1/3} + a_3 T_9^{\,1/3} + a_4 T_9 + a_5 T_9^{\,5/3} + a_6 \ln T_9 \right)$$

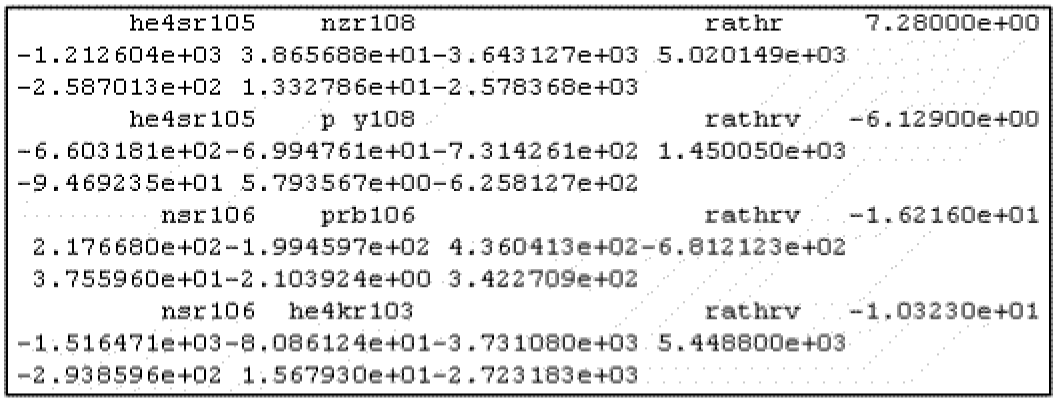
where the temperature T_9 is given in units of 10^9 K [11]. The reaction rates are parameterized over a temperature range of $0.01 \le T_9 \le 10$.

The first ten sets of rates listed in Table 1.2 are based primarily on experimental results. To achieve sufficient accuracy, many experimental rates are split into resonant and non-resonant components (denoted by an “r” or “n” after the reaction source listing). The total reaction rate is the sum of all components contributing to it.

The theoretical rates, *fkth* and *rath*, are calculated using the statistical model programs SMOKER and its successor NON-SMOKER codes, respectively [10]. If a rate is calculated from its inverse via detailed balance, then a “v” is attached to the source abbreviation.

Table 1.1 The eight reaction classifications used in REACLIB

Reaction	Description
a -> b	essentially beta-decays and electron captures
a -> b + c	mainly photodisintegrations or beta-delayed neutron emission
a -> b + c + d	like inverse triple-alpha or beta-delayed two neutron emission
a + b -> c	capture reactions
a + b -> c + d	particle exchange like (p, n)
a + b -> c + d + e	
a + b -> c + d + e + f	
a + b + c -> d (+e)	three particle reactions like triple-alpha



```
       he4sr105    nzr108                        rathr     7.28000e+00
-1.212604e+03 3.865688e+01-3.643127e+03 5.020149e+03
-2.587013e+02 1.332786e+01-2.578368e+03
       he4sr105    p y108                        rathrv   -6.12900e+00
-6.603181e+02-6.994761e+01-7.314261e+02 1.450050e+03
-9.469235e+01 5.793567e+00-6.258127e+02
         nsr106    prb106                        rathrv   -1.62160e+01
 2.176680e+02-1.994597e+02 4.360413e+02-6.812123e+02
 3.755960e+01-2.103924e+00 3.422709e+02
         nsr106  he4kr103                        rathrv   -1.03230e+01
-1.516471e+03-8.086124e+01-3.731080e+03 5.448800e+03
-2.938596e+02 1.567930e+01-2.723183e+03
```

Figure 1.5 A sample set of reactions from REACLIB

Table 1.2 A list of abbreviations used in REACLIB to denote reaction rate source

Abbreviation	Reaction Rate Source
cf88	Caughlan and Fowler 1988
wies	M. Wiescher, J. Goerres, K. Langanke, T. Rauscher, F. Thielemann
laur	L. Wormer, Wiescher, Goerres, Iliades, Thielemann 1994
bb92	Rauscher, Applegate, Cowan, Thielemann, Wiescher 1994
baka	Bao and Kaeppeler 1987
rolf	C. Rolfs
wawo	Wallace and Woosley 1980
mafo	Malaney and Fowler 1988
wfh	Wagoner, Fowler, Hoyle 1964
wag	Wagoner 1969
fkth	Thieleman, Arnould, Truran 1987
rath	Rauscher and Thielemann 2000
btyk	Takahashi, Yamada, Kondo 1973, 1980
bkmo	Klapdor, Metzinger, Oda 1984
mo92	Moeller 1992

Chapter 2

Nucleosynthesis in Astrophysical Phenomena

2.1 The Big Bang

Earlier than one second after the Big Bang, the reversible reactions $p(e^-,\upsilon)n$ and $n(e^+,\overline{\upsilon})p$ maintain the neutron to proton ratio in thermal equilibrium. After about one second, the temperature decreases due to expansion, and the neutron to proton ratio *freezes out* at about 1 to 6. Immediately after this period, the only reaction that changes the number of neutrons is neutron *beta decay*, which is written as $n(e^-\overline{\upsilon})p$.

Without further reactions, the amount of neutrons would quickly decrease, but at a temperature of 10^9 K deuteron formation begins through the reaction $n(p,\gamma)d$. Once deuteron formation has occurred, further reactions proceed to produce helium. Both ^{3}He and ^{4}He are made, along with the radioactive form of hydrogen (^{3}H). Reactions such as $d(n,\gamma)^3He$, $^3H(p,\gamma)^4He$, $d(p,\gamma)^3He$, and $^3He(n,\gamma)^4He$ occur, but they are relatively slow due to the photon emission. Reactions that produce helium at a more rapid pace are $d(d,n)^3He$, $d(d,p)^3H$, $^3H(d,n)^4He$, and $^3He(d,p)^4He$.

Eventually the temperature drops sufficiently that the electrostatic repulsion of the deuterons causes the reactions to stop, and almost all of the neutrons in the Universe wind up in helium-4 nuclei. Extremely small amounts of ^{7}Li and ^{7}Be are also produced. There are no stable isotopes with atomic masses of 5 or 8. This creates a bottleneck preventing the Big Bang from producing elements heavier than mass 8.

2.2 Stellar Nucleosynthesis

2.2.1 The Proton-Proton Chains

In the core of our Sun and similar stars, three chains of nuclear reactions called the proton-proton (PP) chains act to convert hydrogen into helium [4]. The first chain, denoted PP I, can written as $p(p,e^{+}\upsilon_{e})d(p,\gamma)\,^{3}He(^{3}He,2p)\,^{4}He$.

As illustrated in Figure 2.1 [7], the PP I chain involves intermediate reactions that produce deuterium and helium. Each reaction in PP I has a different reaction rate as seen in the figure. The first reaction, which involves the transformation of a proton into a neutron via the weak nuclear force, and the lifetime against a pp fusion reaction is typically 10^{9} years. The second and third reactions require on average one second and one million years to occur, respectively.

The extreme temperatures and pressures within the Sun creates competition between the last step of the PP I chain and the first reaction of the PP II chain. If the helium-3 nuclei (^{3}He) interact with the helium-4 nuclei (^{4}He), the PP II chain of reactions takes place, which can be written as $^{3}He(\alpha,\gamma)^{7}Be(e^{-}\upsilon)^{7}Li(p,\alpha)^{4}He$. In the core of the solar interior, the PP I chain occurs approximately 69% of the time and the PP II chain takes place 31% of the time.

At greater temperatures, the PP II chain begins to compete with the PP III chain as shown in Figure 2.2. The PP III chain involves a proton capture by a beryllium-7 nucleus and can be written as $^{7}Be(p,\gamma)\,^{8}B(e^{+}\upsilon_{e})\,^{8}Be(^{4}He)^{4}He$. In the Sun, only 0.3% of the beryllium-7 nuclei will undergo the PP III chain. The neutrinos from the solar burning reactions are detected in terrestrial detectors, and the “Solar Neutrino Problem” is the discrepancy between predictions of the standard solar model with the standard particle physics model (i.e., massless neutrinos) and the observations [12].

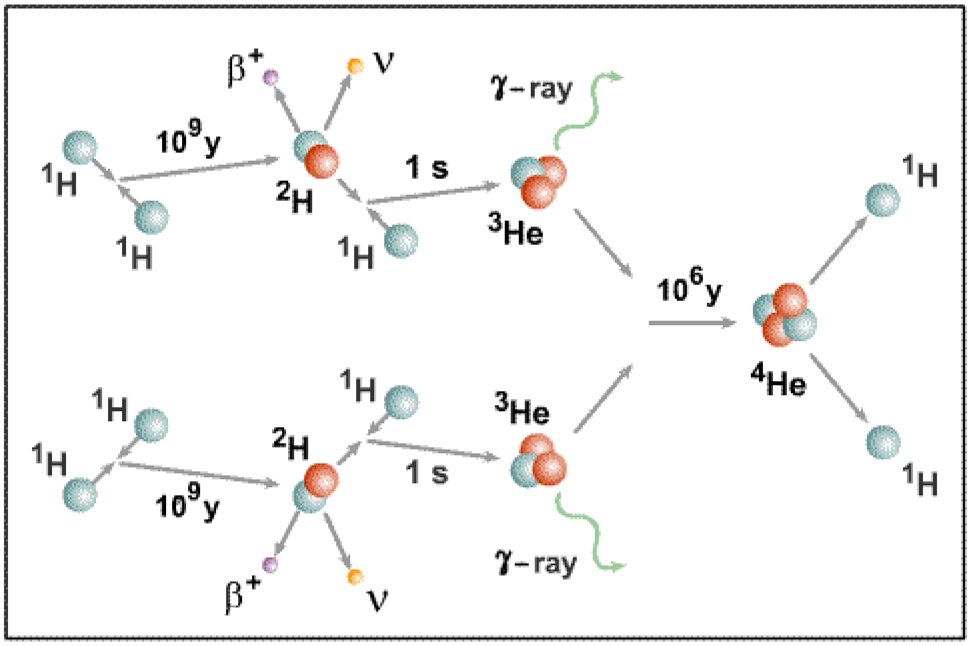


Figure 2.1 A schematic of the PP I chain

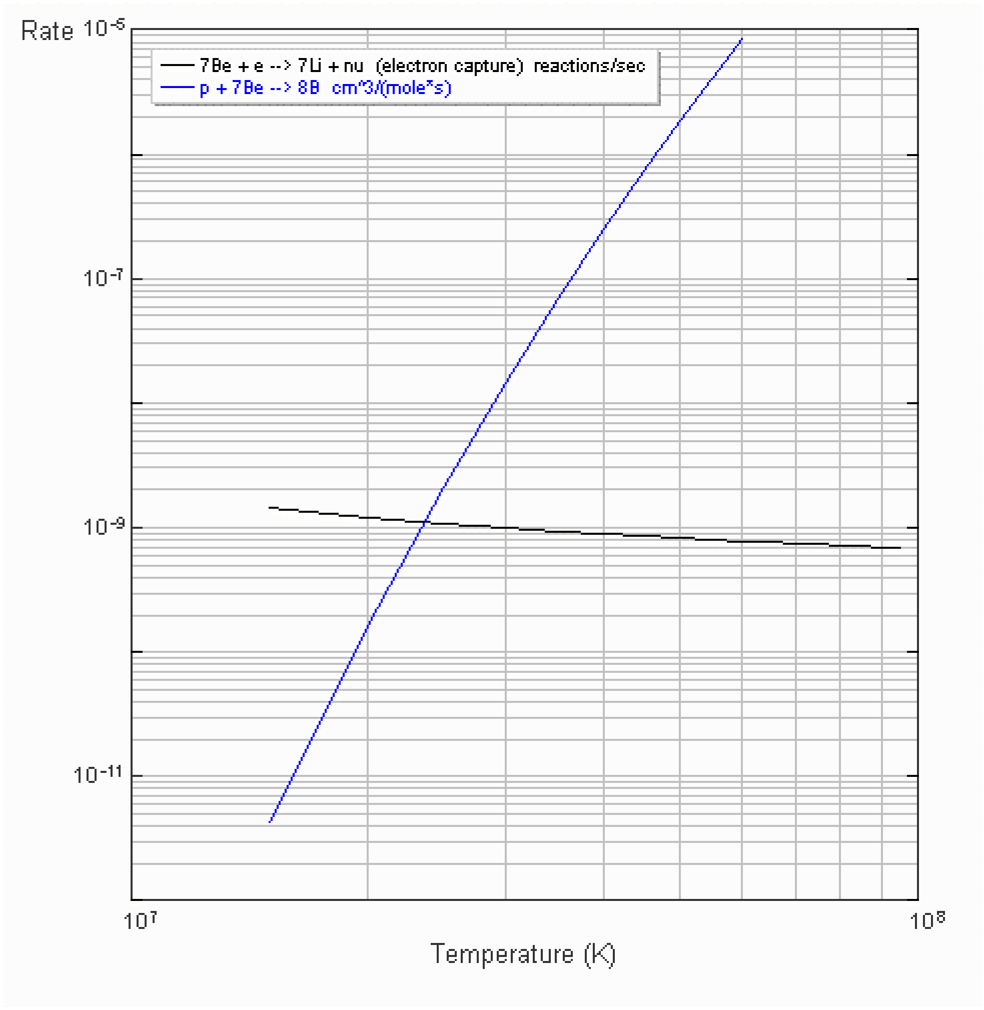


Figure 2.2 A plot illustrating competition between $^7Be(e^-\upsilon)^7Li$ of the PP II chain and $^7Be(p,\gamma)^8B$ of the PP III chain. This plot was constructed using the aLIVe! software.

2.2.2 The CNO Cycle

Another set of reactions, the carbon, nitrogen, and oxygen (CNO) cycle, can catalytically produce ^{4}He from hydrogen. Higher temperatures and densities are required for the CNO cycle to dominate. As seen in Figure 2.3, the temperature of the core regions of our Sun is just below the point where energy production by the PP chains is overtaken by energy production of the CNO cycle [7]. Also note that the CNO cycle has a much stronger temperature dependence than the PP chains. This implies that more massive stars, which are hotter at the core, will have a greater amount of helium produced by the CNO cycles than low-mass stars, which use the PP chains to produce helium at lower temperatures. The most important reactions that compose the CNO cycle are $^{12}C(p,\gamma)^{13}N(e^+\upsilon_e)^{13}C(p,\gamma)^{14}N(p,\gamma)^{15}O(e^+\upsilon_e)^{15}N(p,\alpha)^{12}C$ and are illustrated in Figure 2.4 [7].

2.2.3 Helium Burning Processes

The triple alpha process is a helium burning process that is responsible for carbon production. It involves two reactions: $^4He(\alpha)\,^8Be$ (reversible) and $^8Be(\alpha,\gamma)^{12}C$. The production of beryllium nuclei results in the triple alpha process only after an immediate interaction with the third alpha particle as seen in Figure 2.5 [7]. The second reaction must occur almost immediately after the first because ^{8}Be is unstable with a very short (~10^{-16} sec) lifetime [12]. The triple alpha process is therefore often thought of as a three-body interaction. The temperature dependence of the triple alpha process is even greater than the PP chains or the CNO cycles.

In massive stars, because of the higher temperatures, the helium burning may continue to produce ^{16}O and on to ^{20}Ne in the reactions $^{12}C(\alpha,\gamma)^{16}O(\alpha,\gamma)^{20}Ne$. Alpha particles may continue to create even heavier nuclei until the Coulomb barrier brings the process to a halt near the iron peak region.

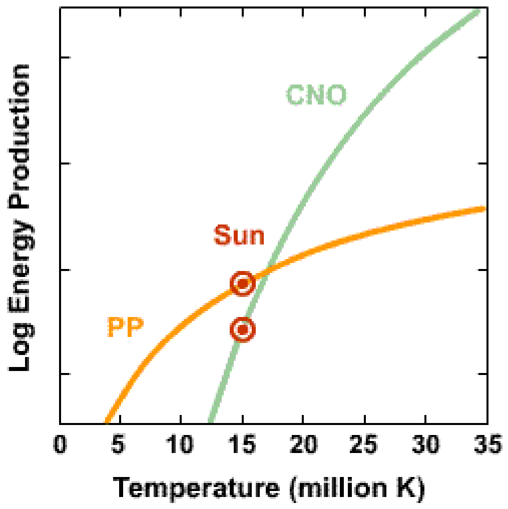


Figure 2.3 A plot of total energy production versus temperature of the PP chains and the CNO cycle. The current temperature at the core of the Sun is also shown.

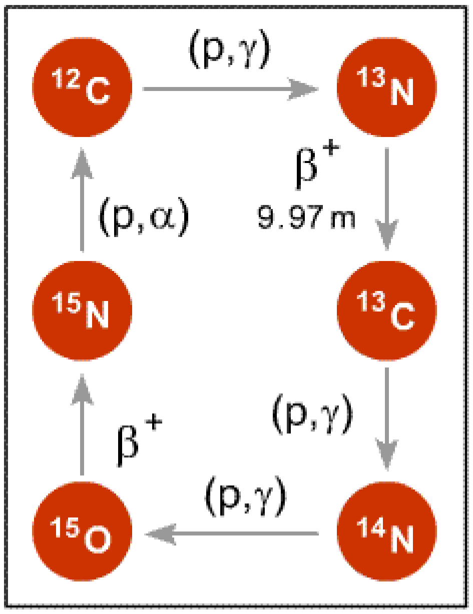


Figure 2.4 A schematic of the CNO cycle

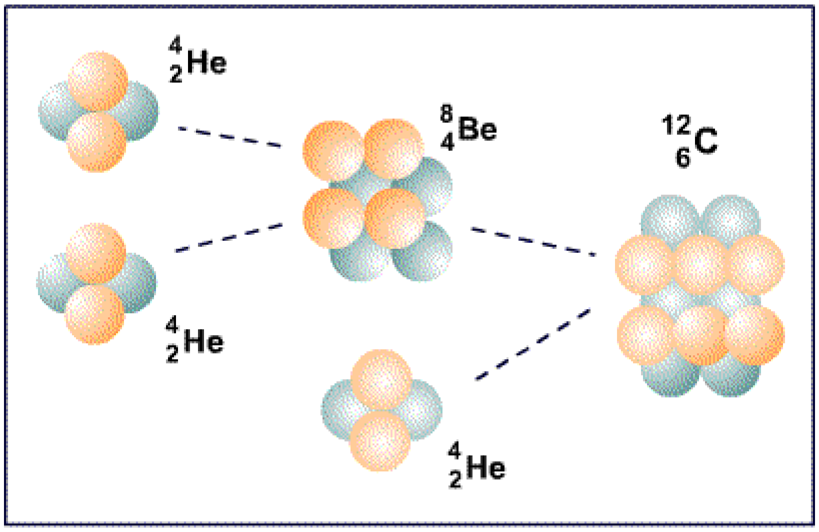


Figure 2.5 A schematic of the triple alpha process

2.2.4 The s-Process

For very massive stars fusion of heavier elements can proceed up to the production of iron just outside of the stellar core. Because of their large coulomb barriers, heavier nuclei don't capture protons easily, so the neutron capture is the primary production channel. The s-process is a process of neutron captures that is responsible for production of roughly half of the heavy elements up to Bi. The s-process refers to a sequence of slow neutron capture reactions (due to a low neutron flux) combined with beta decay creating heavier elements as shown in Figure 2.6 [7]. The schematic shows a progression of neutron captures taking ^{56}Fe to ^{59}Fe. This series of neutron capture reactions ends with a beta decay to ^{59}Co. The neutron capture chain will march through the stable isotopes of an element until it is sufficiently unstable and is likely to beta decay before it can capture another neutron. Thus the s-process is limited to isotopes that lie in or adjacent to the valley of stability.

2.3 Type II Supernova

2.3.1 Overview

Stars with a mass greater than eight solar masses reach a state in which the core of the star resembles an onion. Figure 2.7 displays a schematic of this structure [7]. Successive burning of carbon, oxygen, neon, and silicon has left the star with an iron core that can produce no more nuclear energy. Since the iron core is no longer producing energy, it must be supported by electron degeneracy. At the high temperatures and densities in the core, highly energetic photons destroy the heavy nuclei, producing large amounts of neutrons in a process called photodisintegration.

At this point, free electrons are captured by the protons released during photodisintegration, forming even more neutrons and neutrinos. Since the core has lost large amounts of energy to photodisintegration and the amount of free electrons has decreased, it begins to quickly collapse until densities so extreme that the strong nuclear

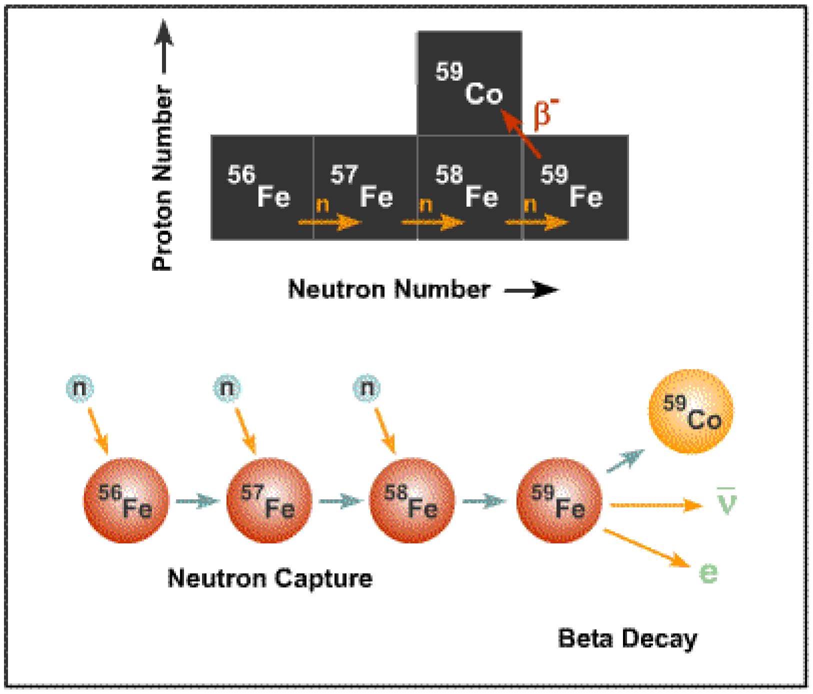


Figure 2.6 A schematic of the s-process

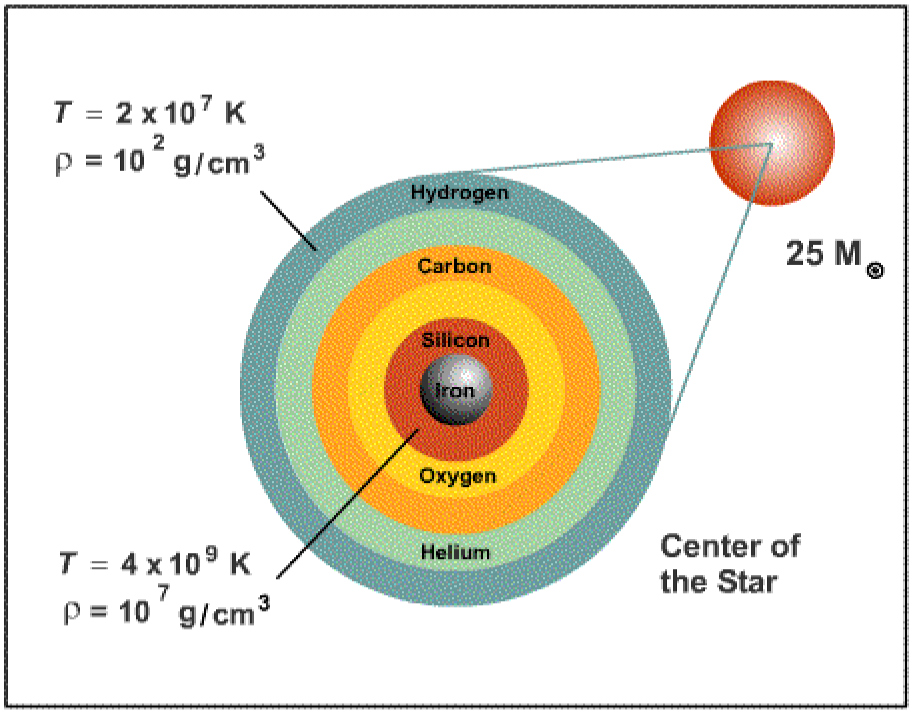


Figure 2.7 The center of a 25 solar mass star late in its life

force becomes repulsive. The inner core rebounds, creating a shockwave and sending a large wave of neutrinos throughout the outer core and the outer layers of matter. It is thought that the r-process may occur in this high entropy, high temperature region that is flooded with a neutrino "wind".

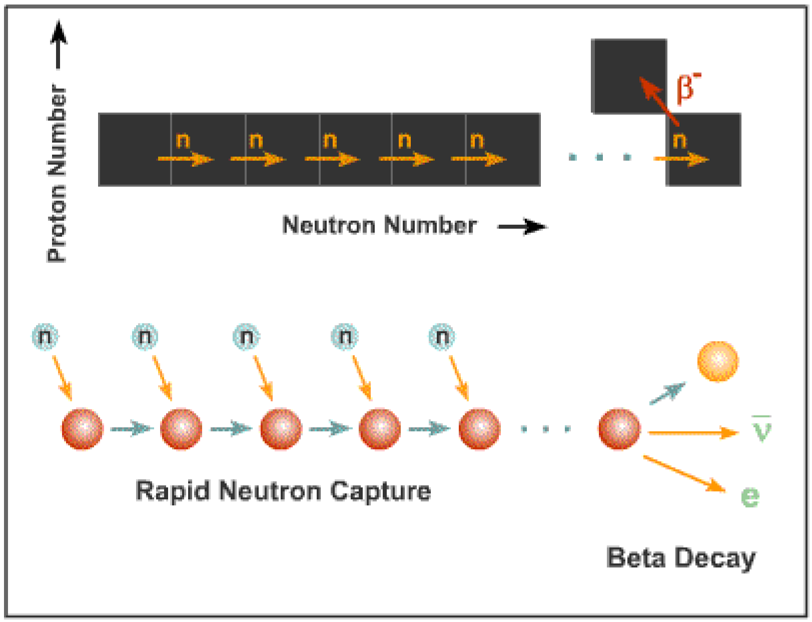
2.3.2 The r-Process

The r-process occurs when the density of free neutrons is extremely high. So high, in fact, that a nucleus undergoing the r-process may absorb 20 neutrons or more before beta decay can take place. These are *rapid* neutron captures. Figure 2.8 shows a schematic of the r-process [7]. Here the lifetime of neutron capture can be seconds or less, smaller than the beta decay lifetimes of many neutron-rich unstable nuclei. The r-process pushes the isotope's neutron number over the edge of the chart of nuclides until beta decay can take place.

2.4 Explosive Hydrogen Burning

2.4.1 The Hot CNO Cycle

Some stars suddenly increase their brightness by great amounts over a period of days, and then slowly dim over a period of months. This increase can be as large as factors of a million. We call such a star a *nova*. A nova occurs in a binary system in which one star is a normal star and one is a white dwarf. Matter from the more normal star accretes in a thin layer on the surface of the white dwarf, either because of the companion star filling its gravitational equipotential and spilling matter onto the white dwarf (typically through an accretion disk), or because of a strong wind from the companion star that the white dwarf captures onto its surface. Eventually this layer may ignite in a thermonuclear explosion. The resulting thermonuclear runaway expels a surface layer of $\sim 10^{-3}$ solar masses into space, while causing a large rise in light output from the system. The mechanism for a nova outburst is illustrated schematically in Figure 2.9 [7].



2.8 A schematic of the r-process

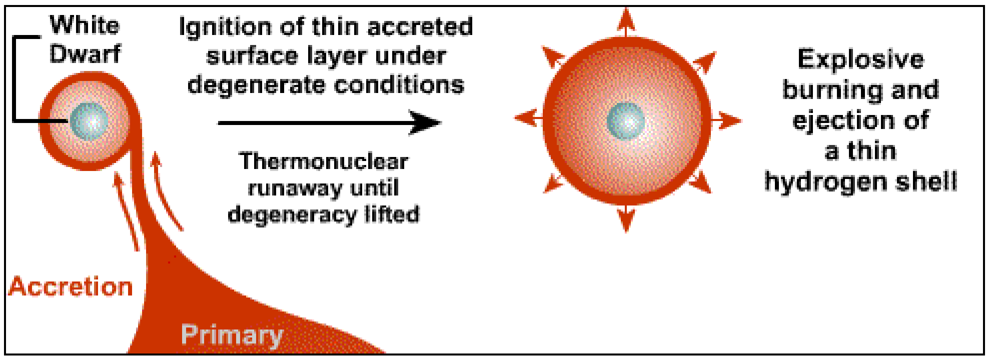

White Dwarf
Ignition of thin accreted surface layer under degenerate conditions
Thermonuclear runaway until degeneracy lifted
Explosive burning and ejection of a thin hydrogen shell
Accretion
Primary

Figure 2.9 A schematic of the nova mechanism

In an X-ray burst, the mechanism is thought to be similar to that for a nova, except that the accretion is onto a neutron star. The X-ray burst is triggered by a thermonuclear runaway under degenerate conditions, as for a nova. However, the gravitational field of a neutron star is much stronger than that of a white dwarf. Matter falling toward the surface of the neutron star is accelerated to high velocities and creates an environment of temperatures and densities greater than that of the nova, which triggers a thermonuclear runaway. This in turn tends to produce X-rays rather than visible light in the thermonuclear runaway. Figures 2.10 and 2.11 illustrate the possible mechanism for the X-ray burst [7].

At these higher temperatures and densities the hot CNO cycle occurs. Figure 2.12 illustrates schematically the transition from the CNO cycle to the hot CNO cycle in the NZ-plane. Figure 2.13 shows the event graphically by plotting reaction rate versus temperature of the ^{13}N breakout reactions. Above a temperature of 0.2 GK, the (p,γ) reaction dominates over the beta decay. Please note that in comparing a beta-decay reaction with a two-particle reaction we are not reflecting the effect of the density factors that must be included to calculate the two-particle reaction rate. The hot CNO cycle may take two paths, depending on the ambient temperatures and densities. Figures 2.14 and 2.15 demonstrate this process. The ^{14}O nuclei may either beta-decay or fuse with helium-4 nuclei to produce either ^{14}N or ^{17}F. Figure 2.16 illustrates the hot CNO cycles in the NZ-plane.

As shown in Figure 2.14 there are three expected opportunities for a breakout from the CNO cycles. The first breakout may occur when ^{18}Ne fuses with a helium-4 nucleus to create a ^{21}Na nucleus instead of beta decaying to ^{18}F. Figure 2.17 illustrates the temperature dependence of this process.

Other possible reactions like the ^{18}F(p, α) ^{15}O and ^{18}F(p, γ) ^{19}Ne reactions are illustrated in Figure 2.14. Figure 2.18 demonstrates the competition between these two reactions. A

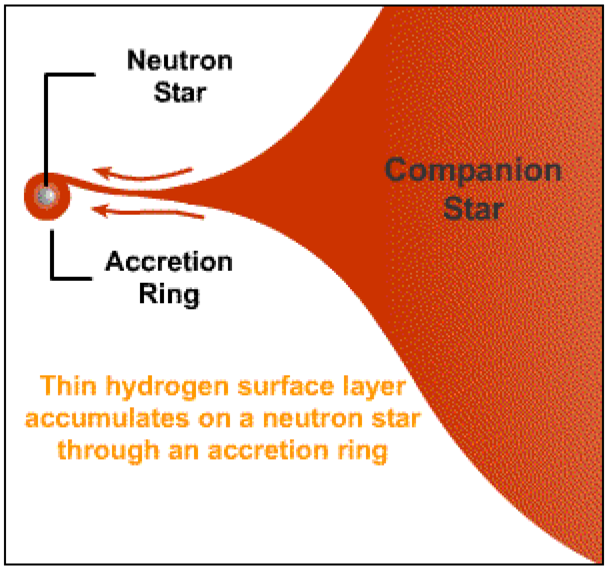

Neutron Star
Companion Star
Accretion Ring
Thin hydrogen surface layer accumulates on a neutron star through an accretion ring

Figure 2.10 A schematic of the accretion of matter onto a neutron star

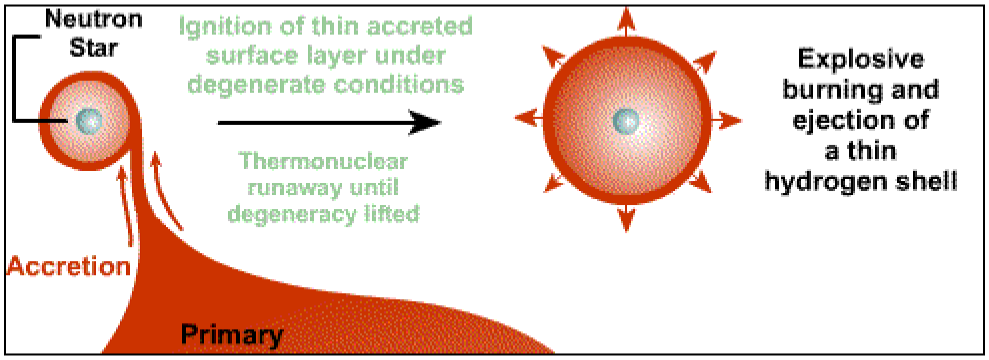

Neutron Star
Ignition of thin accreted surface layer under degenerate conditions
Thermonuclear runaway until degeneracy lifted
Explosive burning and ejection of a thin hydrogen shell
Accretion
Primary

Figure 2.11 A schematic of the X-ray burst mechanism

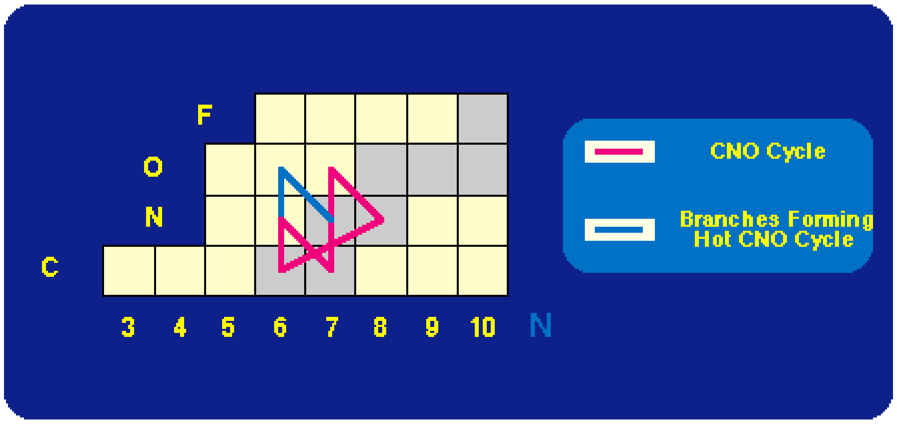


Figure 2.12 A schematic of the CNO and hot CNO cycles in the NZ-plane.

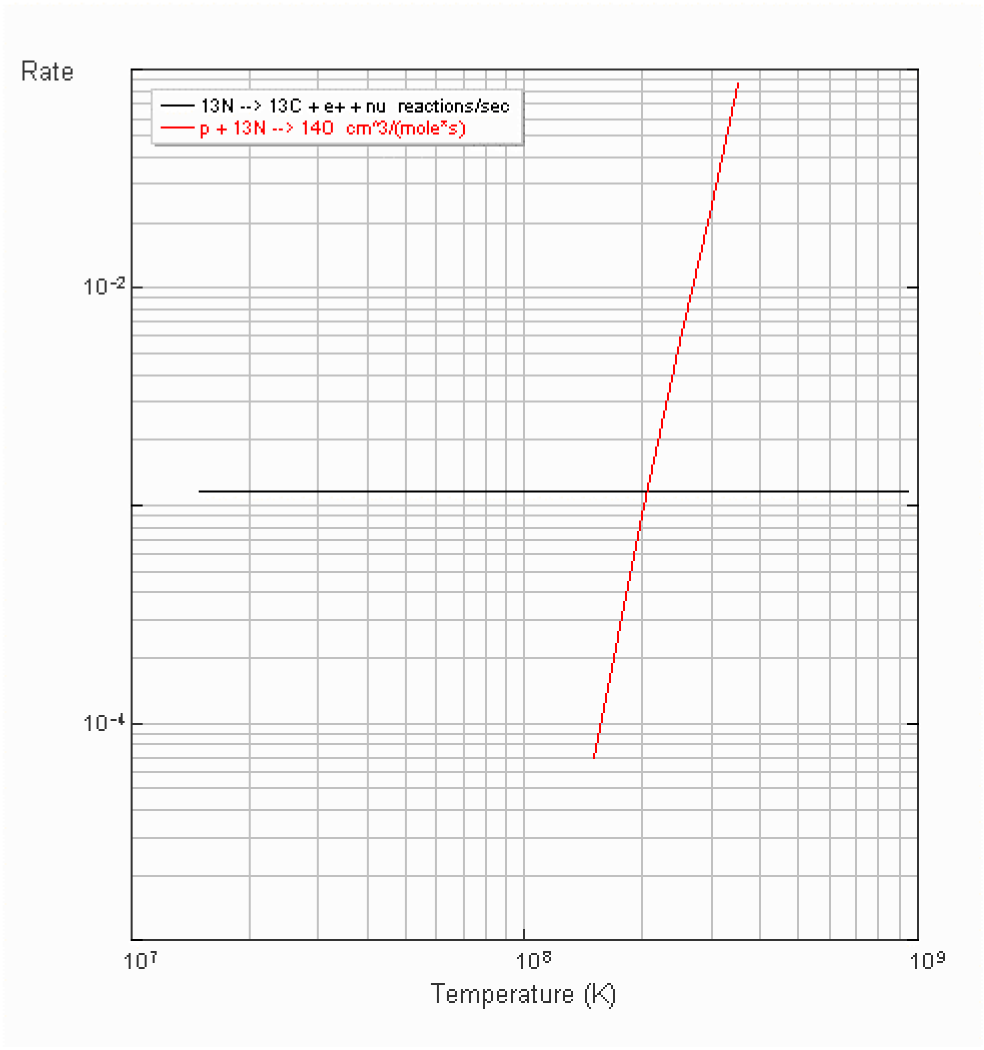


Figure 2.13 A plot illustrating competition between $^{13}N(e^{+}\upsilon_{e})^{13}C$ and $^{13}N(p,\gamma)^{13}C$.

This plot was constructed using the aLIVe! software.

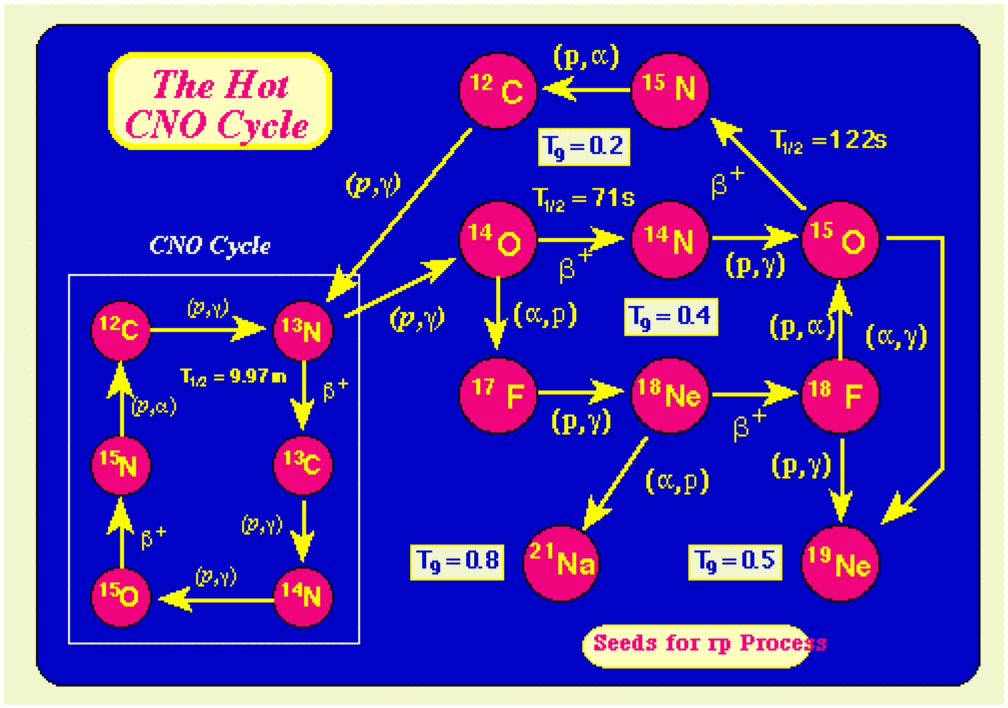


Figure 2.14 A schematic of the CNO and hot CNO cycles. Also shown are the three paths thought to lead to break out of the hot CNO cycle.

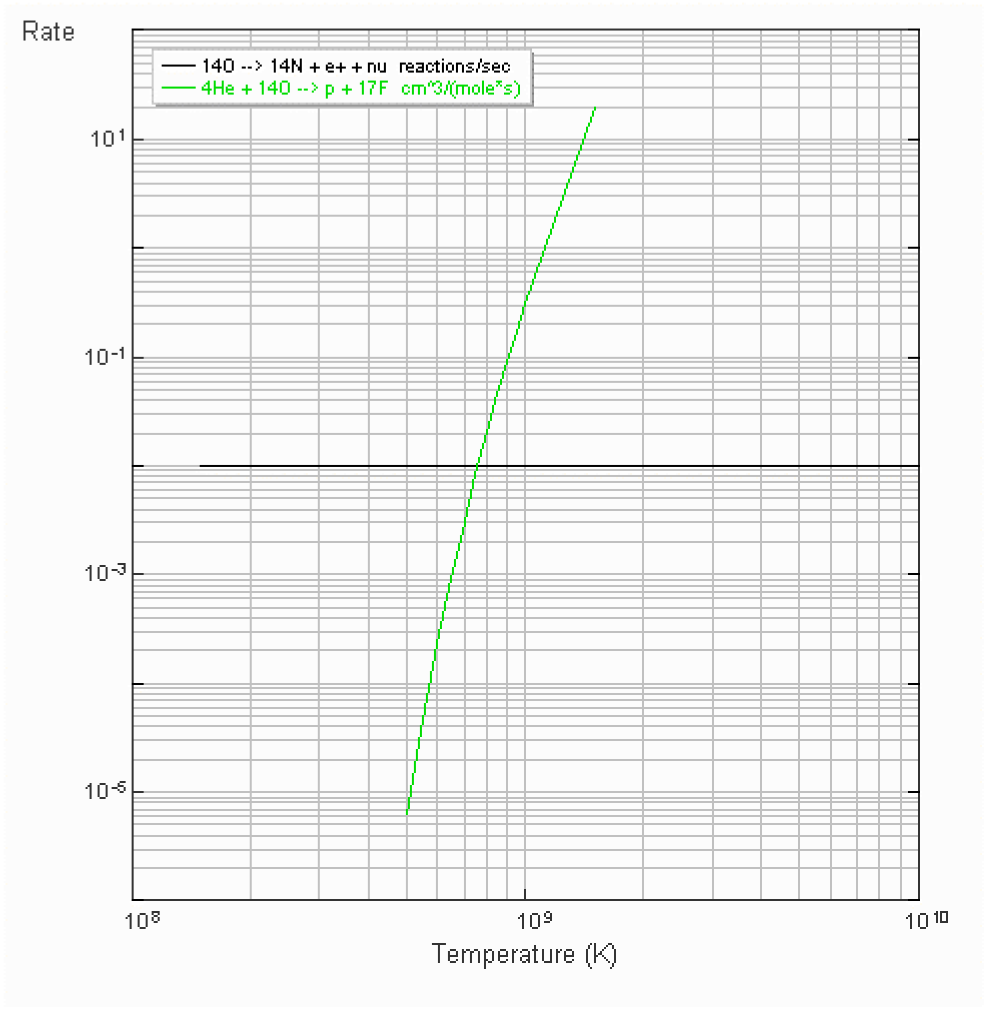


Figure 2.15 A plot illustrating competition between $^{14}O(e^{+}\upsilon_{e})^{14}N$ and $^{14}O(\alpha,p)^{17}F$.

This plot was constructed using the aLIVe! software.

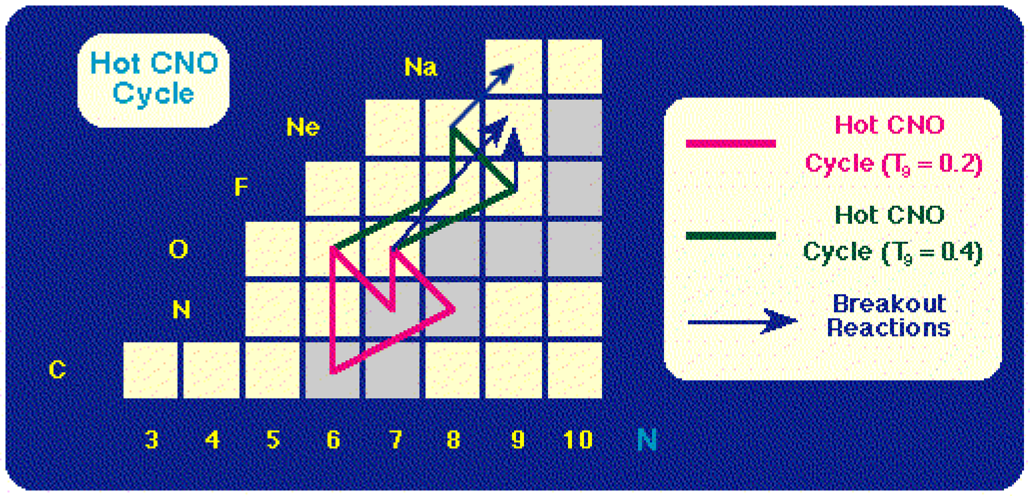


Figure 2.16 Possible hot CNO breakout paths in the NZ-plane.

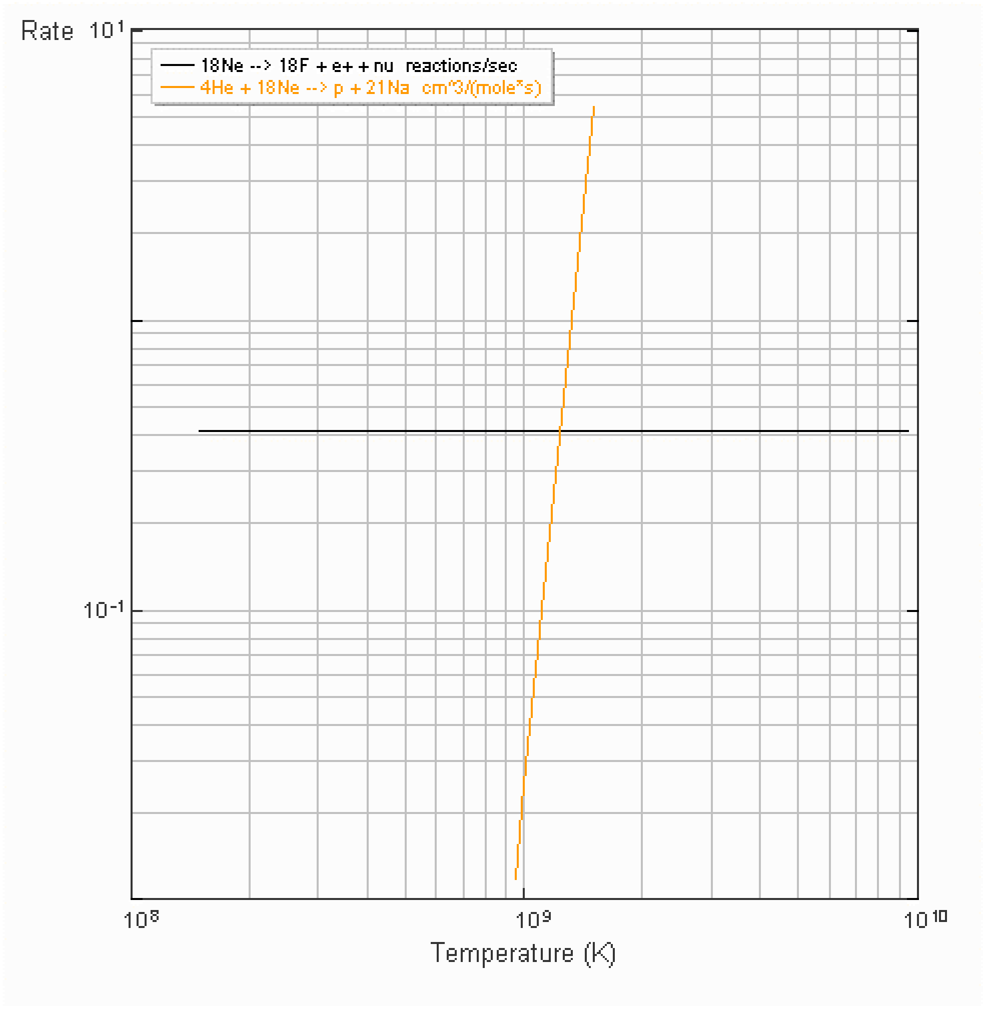


Figure 2.17 A plot illustrating competition between $^{18}Ne(e^{+}\upsilon_{e})^{18}F$ and $^{18}Ne(\alpha,p)^{21}Na$.

This plot was constructed using the aLIVe! software.

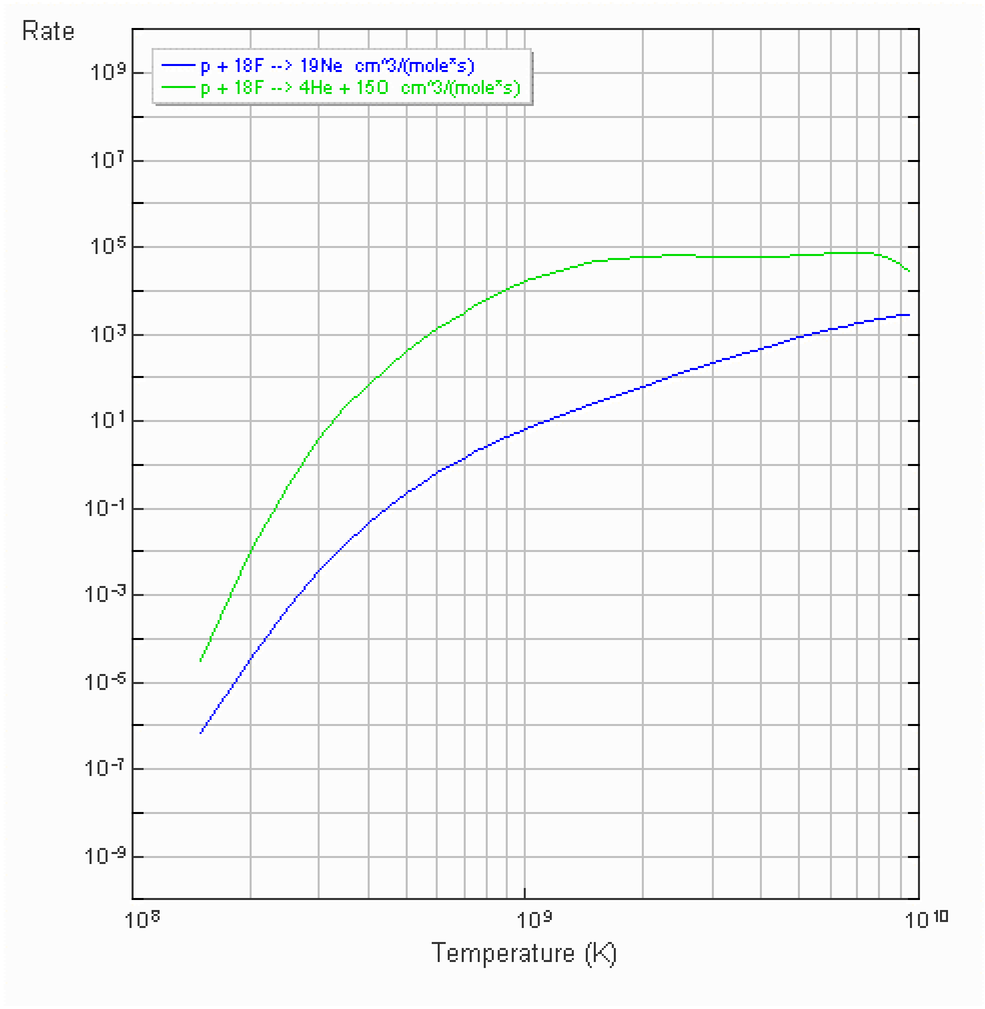


Figure 2.18 A plot illustrating competition between $^{18}F(p,\gamma)^{19}Ne$ and $^{18}F(p,\alpha)^{15}O$.

This plot was constructed using the aLIVe! software.

final possible competition seen in the hot CNO cycles involves the beta decay of the ^{15}O nuclei to ^{15}N or the ^{15}O(α, γ) ^{19}Ne. The reaction rate is plotted versus temperature for the two reactions in Figure 2.19. As we shall see later in this chapter, the hot CNO cycle breakouts are the seeds for the rp-process and αp-process.

In a Master's Thesis by Suzanne Parete-Koon, the effects of a new reaction rate for $^{17}F(p,\gamma)^{18}Ne$ measured at Oak Ridge National Laboratory on nova nucleosynthesis was compared to the elemental abundances calculated by an older rate currently in REACLIB. Figure 2.20 shows the beta-decay rate for ^{17}F in a temperature region characteristic of the nova environment. From the plot we can see that the older REACLIB rate for $^{17}F(p,\gamma)^{18}Ne$ does not begin to compete with the beta-decay rate until a temperature of approximately 0.32 GK, while the contribution of the ORNL rate competes very little with the decay rate at this temperature. The ORNL rate begins to compete with the beta-decay at a temperature of 0.48 GK, but at this temperature the REACLIB rate is several factors larger. Within this temperature range, the REACLIB rate differs by a factor of 30 from the ORNL rate at nova temperatures. Since the temperature range of 1-4 * 10^8 is so crucial for novae, the plot suggests that there may be significant differences when the new ^{17}F(p,γ) rate is used in a nova nucleosynthesis calculation. This was found to be true when the coupled set of non-linear differential equations describing the element synthesis were solved [13].

In a Master's Thesis by Luc Dessieux, the astrophysical implications of the reaction rate of $^{14}O(\alpha,p)^{17}F$ were investigated [14]. At nova temperatures one would expect that this reaction would contribute little when compared to the beta decay of ^{14}O to ^{14}N. In the temperature range between 0.1 and 0.4 GK, the $^{14}O(\alpha,p)^{17}F$ rate is 6 to 9 orders of magnitude smaller than the beta-decay rate as seen in Figure 2.21. This suggests that this reaction does not play crucial role at nova temperatures. In X-ray burst conditions (temperatures between 0.7 and 1.0 GK), the $^{14}O(\alpha,p)^{17}F$ rate begins to compete with

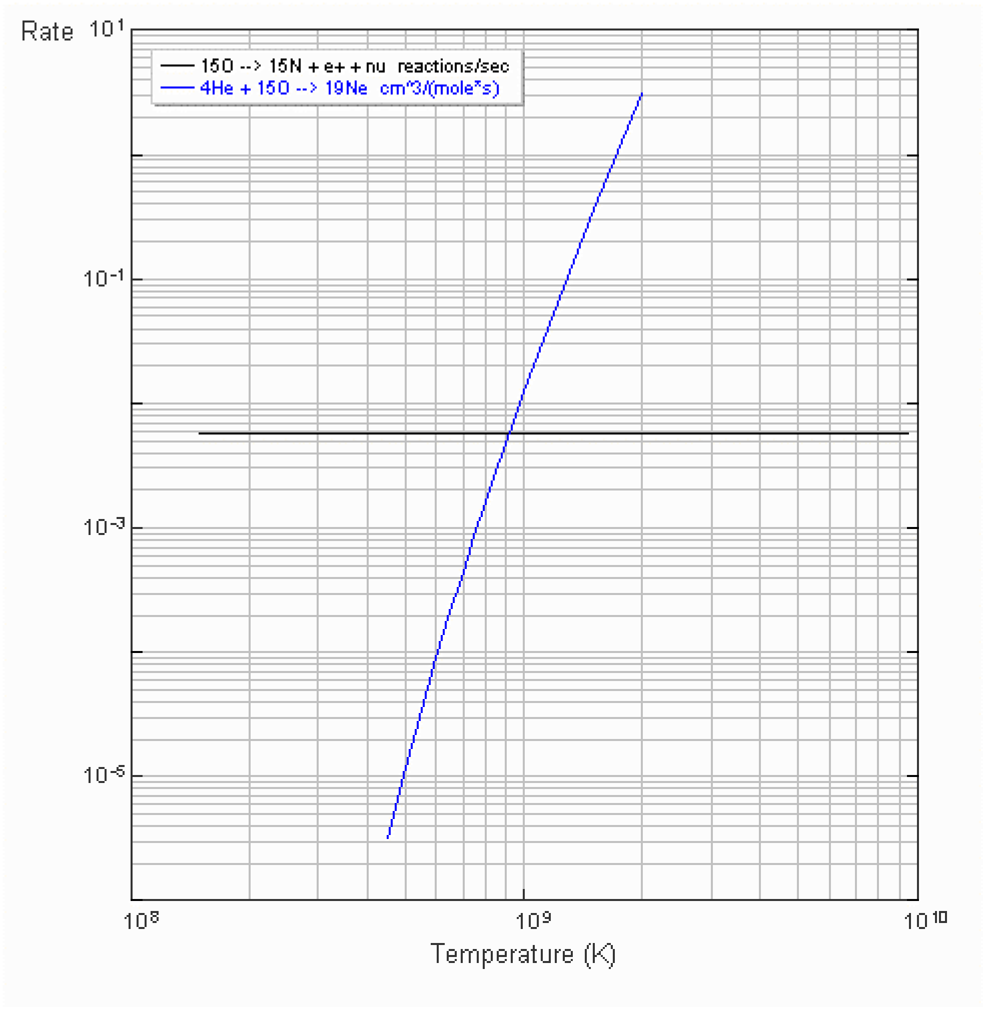


Figure 2.19 A plot illustrating competition between $^{15}O(e^{+}\upsilon_{e})^{15}N$ **and** $^{15}O(\alpha,\gamma)^{19}Ne$.

This plot was constructed using the aLIVe! software.

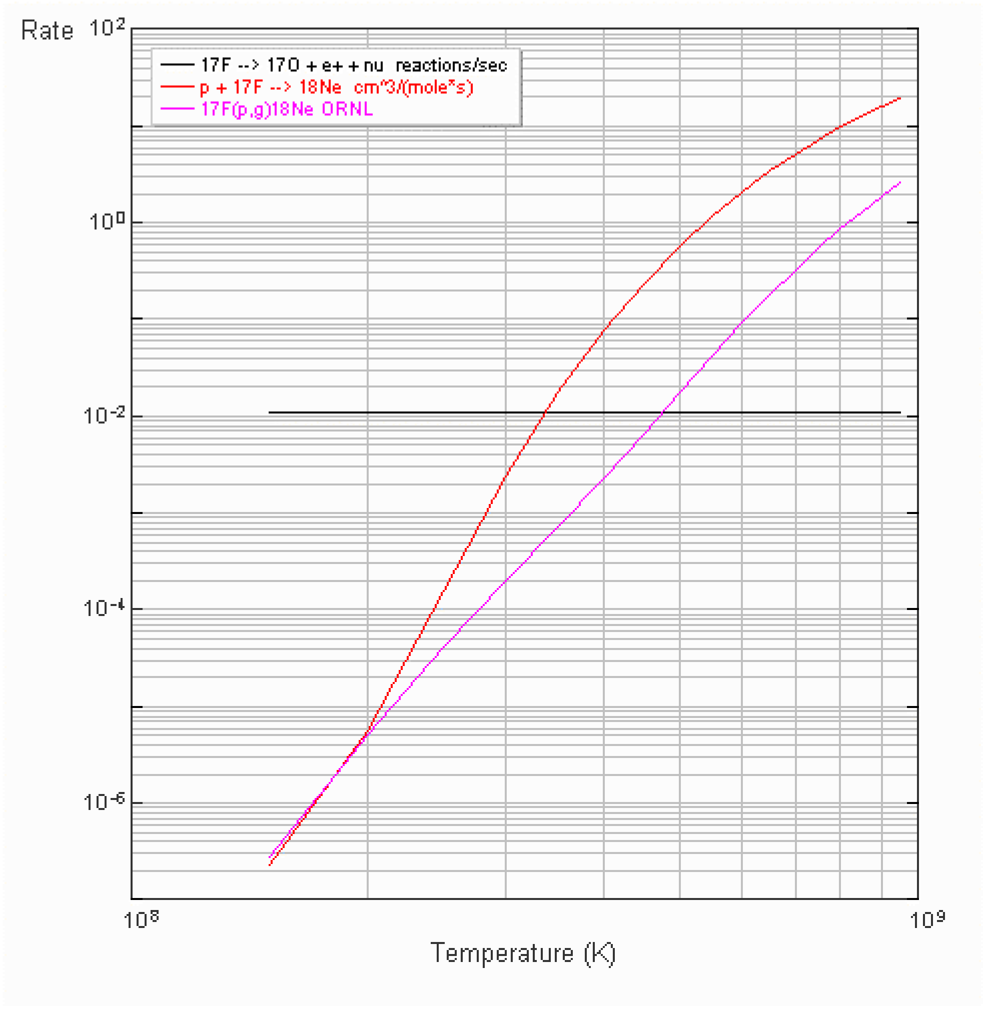


Figure 2.20 A graphical comparison of the reaction rate for $^{17}F(p,\gamma)^{17}O$ as determined by Weischer and ORNL with the $^{17}F(e^{+}\upsilon_{e})^{17}O$ decay rate. This plot was created using the aLIVe! software.

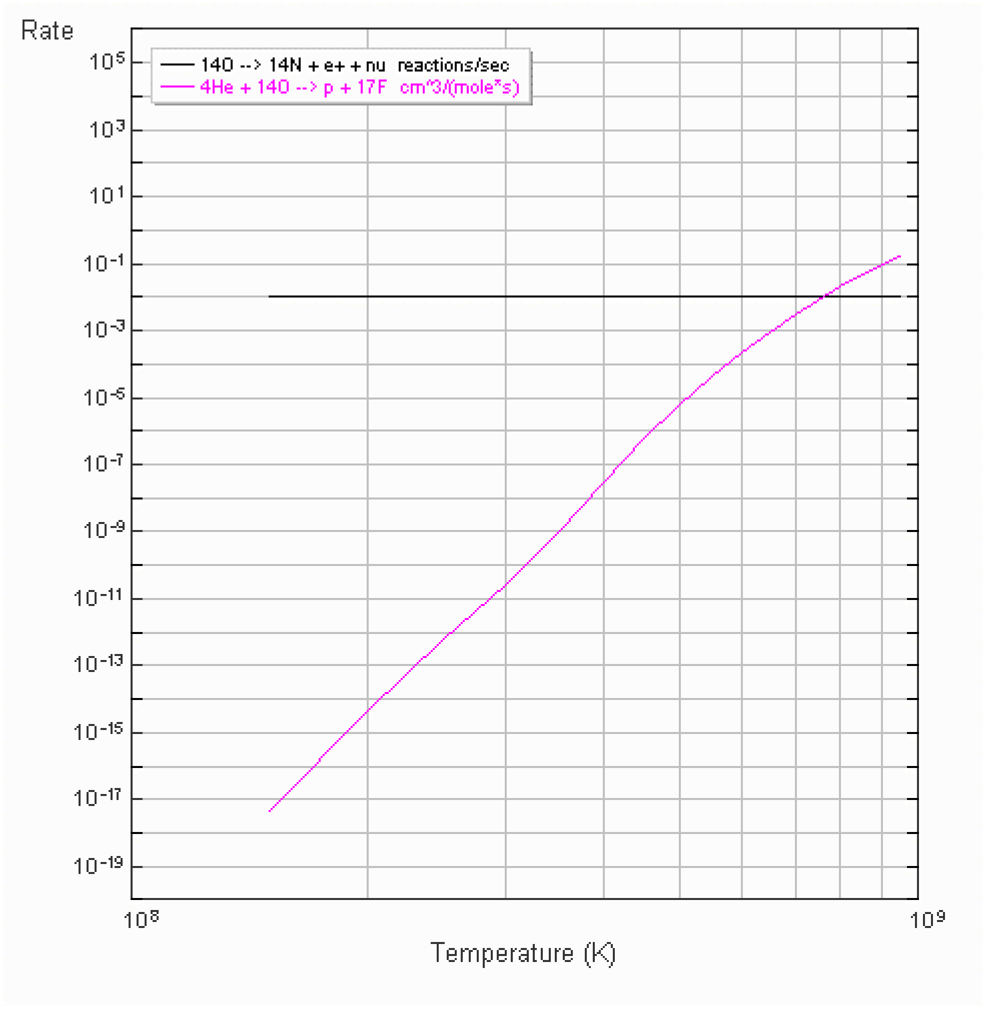


Figure 2.21 A graphical comparison of the reaction rate for $^{14}O(\alpha, p)^{17}F$ with the $^{14}O(e^{+}\upsilon_{e})^{17}F$ decay rate. This plot was created using the aLIVe! software.

and eventually overtakes the beta-decay rate. As shown in Figure 2.14, this suggests that $^{14}O(\alpha, p)^{17}F$ reaction opens a path for nuclei to break out of the hot CNO cycles and possibly begin the rp-process.

2.4.2 The rp-Process

The rp-process is responsible for the production of many proton-rich nuclei and is similar to the r-process except that protons or alpha particles are being captured by the seed nuclei and β^+ decays occur. Figure 2.22 schematically illustrates this process [7]. The primary site of the rp-process is thought to be X-ray bursts and perhaps very hot novae. Figure 2.23 shows the paths along the chart of nuclides for the s-, r-, and rp-processes [7].

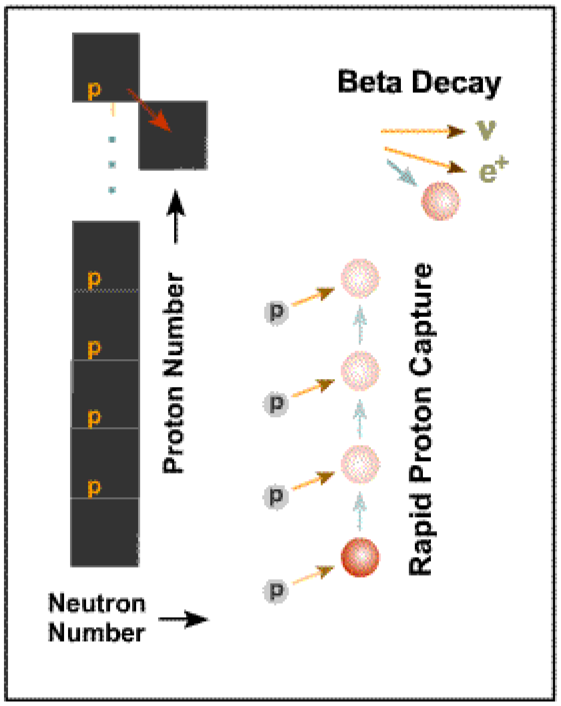


Figure 2.22 A schematic of the rp-process

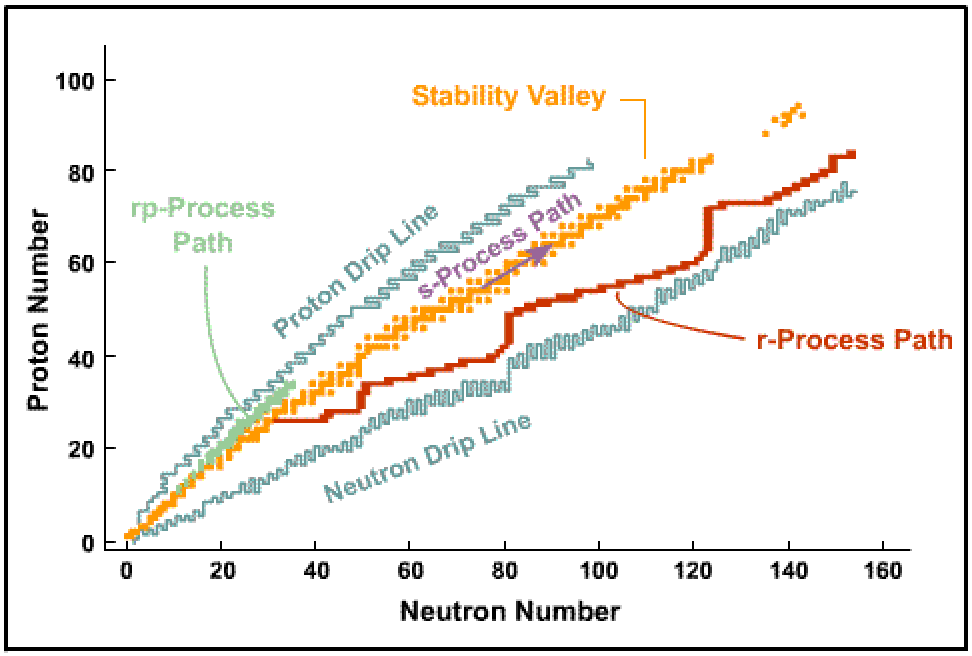


Figure 2.23 The paths of the s-, r-, and rp-processes in the chart of nuclides

Chapter 3

Visualization of Thermonuclear Reaction Rates

3.1 Previous Visualization Initiatives

3.1.1 The Caughlan and Fowler 1988 Implementation

The Caughlan and Fowler 1988 Thermonuclear Reaction Rate Collection, or CF88, was the first major rate library to be posted on the World Wide Web with a complete graphical representation [15]. This website included a graphical interface for choosing isotopes, tables of temperature versus reaction rate values, and corresponding plots. The graphical interface, constructed in HTML, is an interactive chart of the nuclides up to silicon. By clicking on the isotope's representative hyperlink, the user is taken directly to a list of available reactions. Selection of a reaction rate opens a list containing links to tables of values and GIF or postscript plots.

Although this initial distribution of reaction rate data employs modern technology with intuitive user interfaces, the collection is limited by its lack of dynamic content. This is most noticeable for the downloadable plots. Since the plots are preprogrammed, it is impossible for the user to change any characteristic of the plot's format. Properties such as scale, tickmarks, gridlines, and titles are all preset. Also, only one reaction can be viewed on any plot, which inhibits direct comparisons between two or more rates and their curves. Furthermore, the user cannot add any additional plots and compare them to those in the library. Figure 3.1 is an example of such a plot.

3.1.2 The Hauser-Feshbach Java Interface

T. Rauscher's "Astrophysical Cross Sections and Reaction Rates" website employs Java and HTML interfaces to access results of calculations made with the Hauser-Feshbach

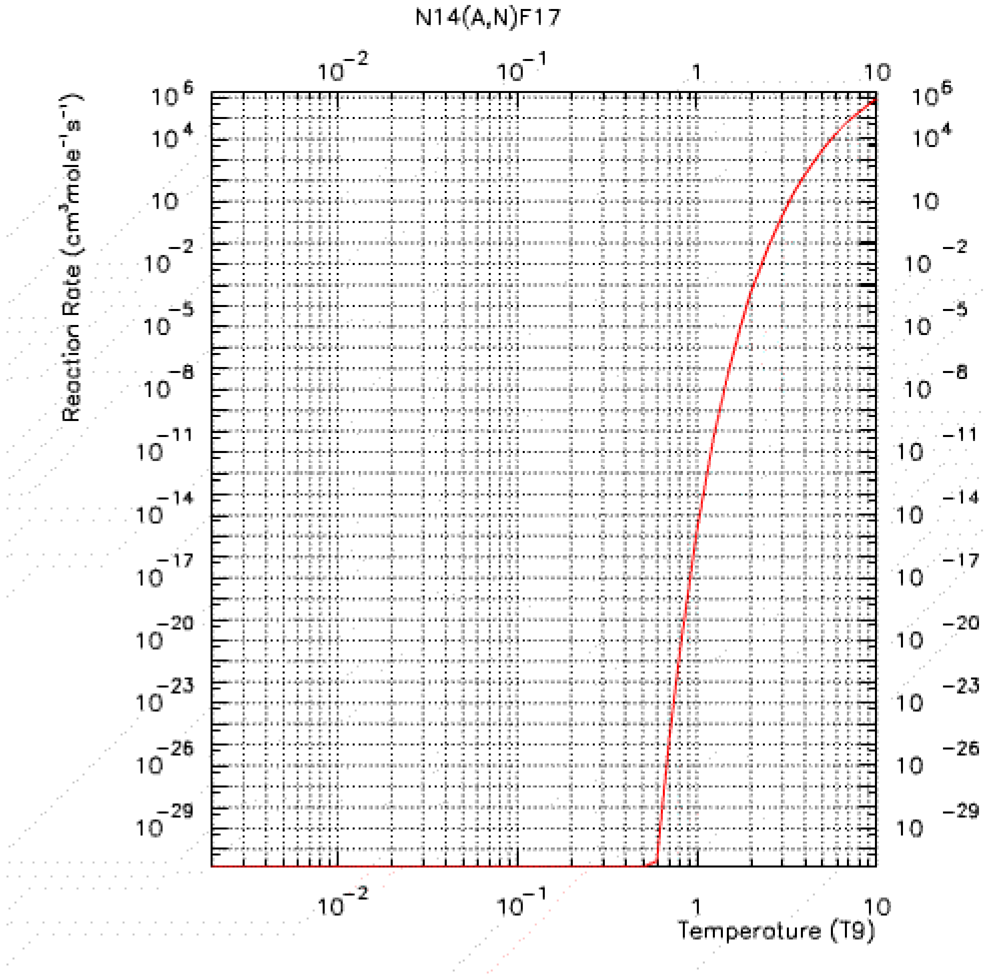


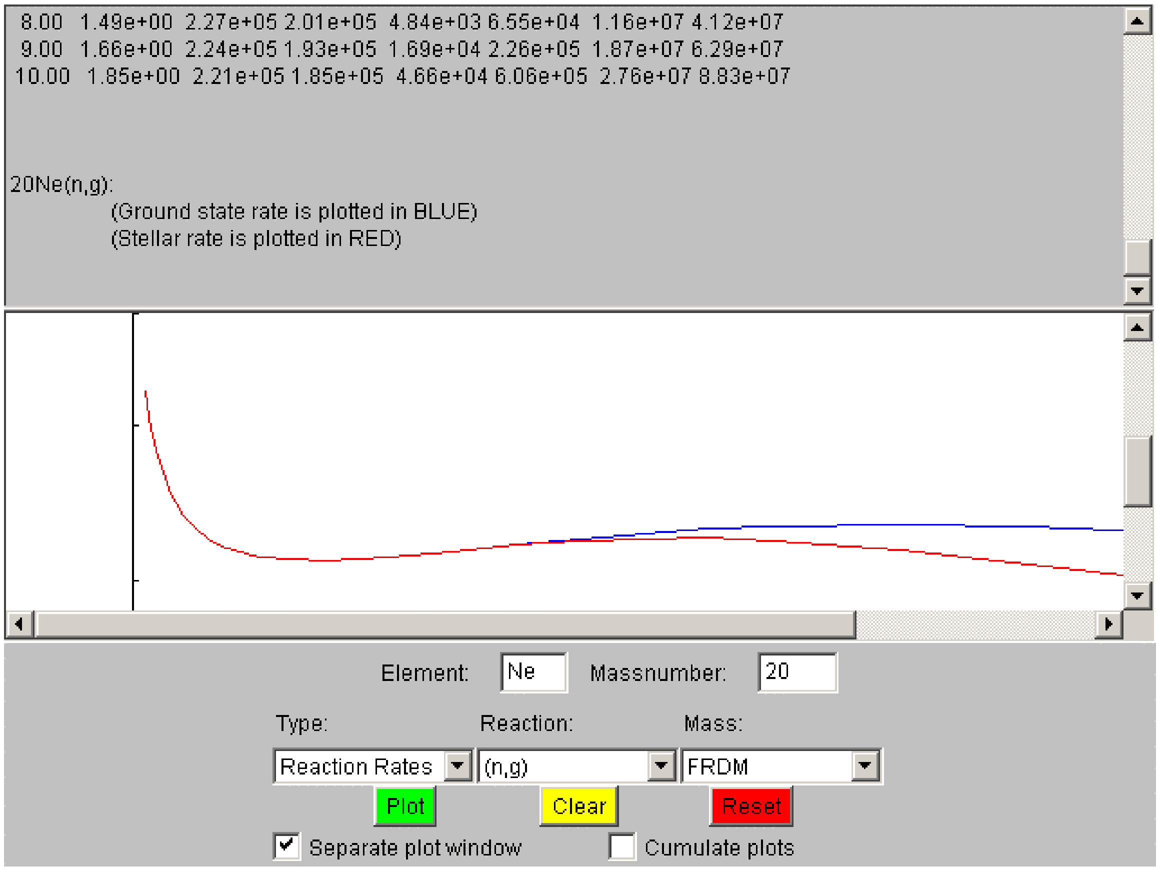
Figure 3. 1 A reaction rate versus temperature plot from CF88

statistical model [16]. Figure 3.2 displays an example query via the Java interface. The interactive elements seen here allow the user to examine reaction rates and cross sections for elements from Ne to Bi. By entering the element's symbol and mass number, the user can view plots for particular reaction types in the center window or in a separate window, as seen in Figure 3.3. The top window displays a set of data including the reaction rate versus temperature points. An important feature of this implementation is the ability to highlight and copy text from the Java applet and paste the text to any system application. The user may then use the data to reproduce the plot elsewhere. However, it lacks the ability to rescale plots, overlay reactions, add reactions, and has no simple graphical user interface. It also works only on theoretical rates.

3.1.3 Interactive Table of the Nuclides with a Cross Section Plotter

The Nuclear Data Evaluation Lab at the Korea Atomic Energy Research Institute implements an interactive table of the nuclides to access a bounty of information quickly by providing atomic and nuclear data at a mouse-click [17]. The expansive chart is split into seven sections. Each isotope is color coded according to half-life as seen in Figure 3.4.

A custom plotter called ENDFPLOT-2 for nuclear cross sections is also available at the site. ENDFPLOT-2 is a CGI program that can generate graphs from the ENDF cross section library. Figure 3.5 shows a plot generated at this site. Several favorable features are implemented in the plotting program. First and foremost is the ability to view more than one curve at a time. The program also has the ability to export the plot in the EPS graphics file format. By choosing the "Text Data" link, a browser window is opened containing the data used to produce the plot. The plot also offers an interactive legend where dropdown menus determine each curve's line color. The user can also change the scales of the energy and cross section axis with the interface shown in Figure 3.6. Dropdown menus are used here to allow a choice between logarithmic and linear



```
 8.00   1.49e+00  2.27e+05 2.01e+05  4.84e+03 6.55e+04  1.16e+07 4.12e+07
 9.00   1.66e+00  2.24e+05 1.93e+05  1.69e+04 2.26e+05  1.87e+07 6.29e+07
10.00   1.85e+00  2.21e+05 1.85e+05  4.66e+04 6.06e+05  2.76e+07 8.83e+07


20Ne(n,g):
              (Ground state rate is plotted in BLUE)
              (Stellar rate is plotted in RED)
```

Element: Ne   Massnumber: 20

Type: Reaction Rates   Reaction: (n,g)   Mass: FRDM

[✓] Separate plot window   [ ] Cumulate plots

Figure 3. 2 T. Rauscher's Java interface for plotting nuclear reaction rates and cross sections

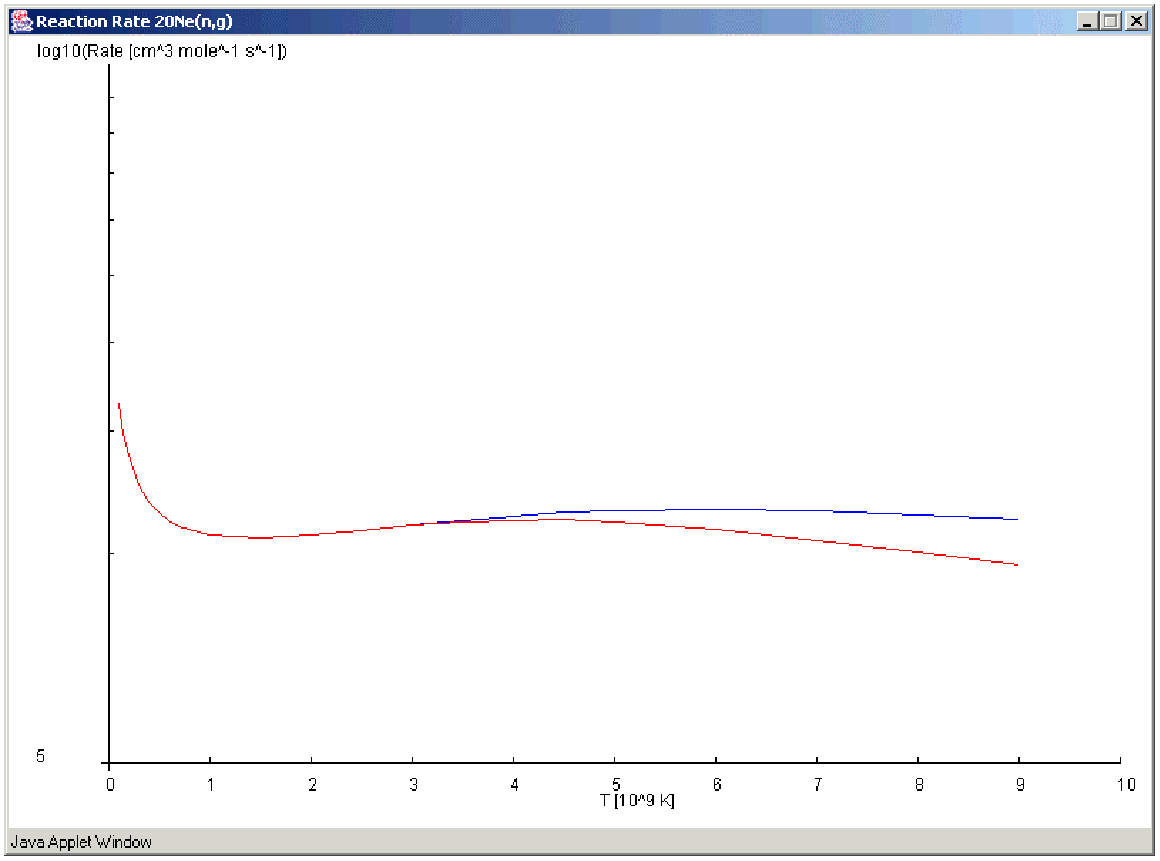
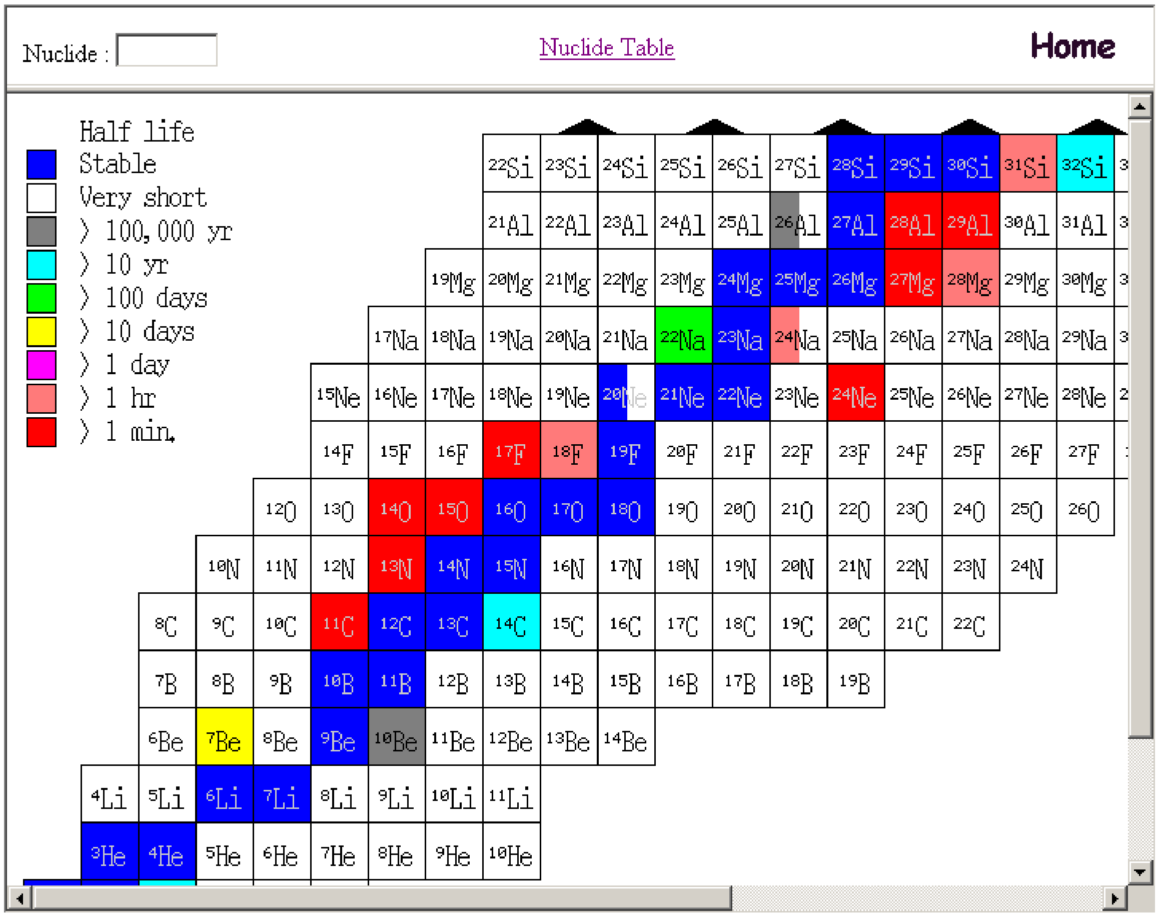


Figure 3. 3 A reaction rate versus temperature plot from the T. Rauscher Java interface



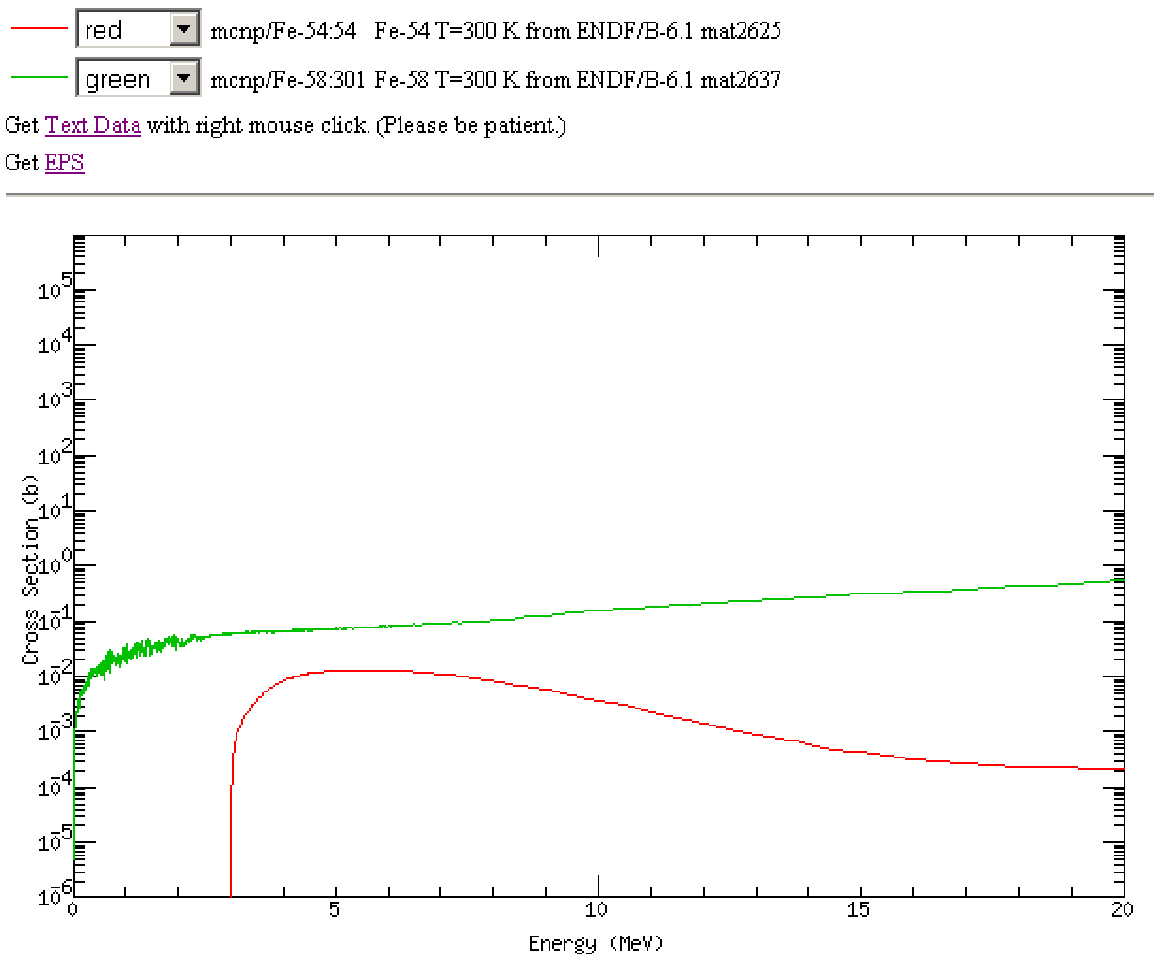
Nuclide : Nuclide Table Home

Half life

🟦	Stable
⬜	Very short
⬛	〉 100, 000 yr
🟦	〉 10 yr
🟩	〉 100 days
🟨	〉 10 days
🟪	〉 1 day
🟥	〉 1 hr
🟥	〉 1 min.

							^{22}Si	^{23}Si	^{24}Si	^{25}Si	^{26}Si	^{27}Si	^{28}Si	^{29}Si	^{30}Si	^{31}Si	^{32}Si	3
							^{21}Al	^{22}Al	^{23}Al	^{24}Al	^{25}Al	^{26}Al	^{27}Al	^{28}Al	^{29}Al	^{30}Al	^{31}Al	3
						^{19}Mg	^{20}Mg	^{21}Mg	^{22}Mg	^{23}Mg	^{24}Mg	^{25}Mg	^{26}Mg	^{27}Mg	^{28}Mg	^{29}Mg	^{30}Mg	3
					^{17}Na	^{18}Na	^{19}Na	^{20}Na	^{21}Na	^{22}Na	^{23}Na	^{24}Na	^{25}Na	^{26}Na	^{27}Na	^{28}Na	^{29}Na	3
				^{15}Ne	^{16}Ne	^{17}Ne	^{18}Ne	^{19}Ne	^{20}Ne	^{21}Ne	^{22}Ne	^{23}Ne	^{24}Ne	^{25}Ne	^{26}Ne	^{27}Ne	^{28}Ne	2
				^{14}F	^{15}F	^{16}F	^{17}F	^{18}F	^{19}F	^{20}F	^{21}F	^{22}F	^{23}F	^{24}F	^{25}F	^{26}F	^{27}F	:
			^{12}O	^{13}O	^{14}O	^{15}O	^{16}O	^{17}O	^{18}O	^{19}O	^{20}O	^{21}O	^{22}O	^{23}O	^{24}O	^{25}O	^{26}O	
		^{10}N	^{11}N	^{12}N	^{13}N	^{14}N	^{15}N	^{16}N	^{17}N	^{18}N	^{19}N	^{20}N	^{21}N	^{22}N	^{23}N	^{24}N		
	^{8}C	^{9}C	^{10}C	^{11}C	^{12}C	^{13}C	^{14}C	^{15}C	^{16}C	^{17}C	^{18}C	^{19}C	^{20}C	^{21}C	^{22}C			
	^{7}B	^{8}B	^{9}B	^{10}B	^{11}B	^{12}B	^{13}B	^{14}B	^{15}B	^{16}B	^{17}B	^{18}B	^{19}B					
	^{6}Be	^{7}Be	^{8}Be	^{9}Be	^{10}Be	^{11}Be	^{12}Be	^{13}Be	^{14}Be									
^{4}Li	^{5}Li	^{6}Li	^{7}Li	^{8}Li	^{9}Li	^{10}Li	^{11}Li											
^{3}He	^{4}He	^{5}He	^{6}He	^{7}He	^{8}He	^{9}He	^{10}He											

Figure 3.4 A portion of the interactive table of the nuclides at The Nuclear Data Evaluation Lab/Korea Atomic Energy Research Institute's website



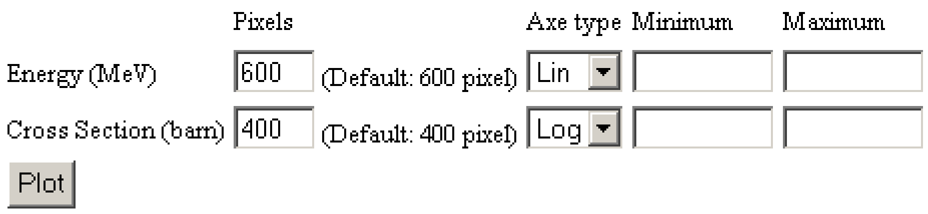
🟥 red mcnp/Fe-54:54 Fe-54 T=300 K from ENDF/B-6.1 mat2625

🟩 green mcnp/Fe-58:301 Fe-58 T=300 K from ENDF/B-6.1 mat2637

Get Text Data with right mouse click. (Please be patient.)

Get EPS

Figure 3.5 A reaction rate versus temperature plot with interactive legend at The Nuclear Data Evaluation Lab/Korea Atomic Energy Research Institute's website



	Pixels		Axe type	Minimum	Maximum
Energy (MeV)	600	(Default: 600 pixel)	Lin		
Cross Section (barn)	400	(Default: 400 pixel)	Log		

Figure 3.6 A set of plotting format options at The Nuclear Data Evaluation Lab/Korea Atomic Energy Research Institute's website

scales for each axis, and text fields are provided to input a maximum and minimum value for each axis. It does not have access to reaction rates, the ability to add new rates, and a simple graphical user interface.

3.2 Motivations

3.2.1 A Need for a New Plotting Package

The REACLIB nuclear reaction rate library is a standard for nuclear astrophysics applications, but has no custom technology to plot and investigate the information within it. We will establish and substantiate a general need for a custom computing environment to plot and analyze nuclear reaction rates in astrophysical applications. In the past, those interested in investigating the content of REACLIB have been forced to employ simple all-purpose graphing tools or spend a large amount of time programming small plotting routines with packaged languages such as MATLAB or IDL. By providing a user-friendly, portable, and compact plotting package for REACLIB, astrophysicists will be able to investigate reaction rates in a standardized format quickly.

3.2.2 Current Technology

Current technologies such as Java are now freely available to the public and provide powerful tools and programming components to quickly construct custom analysis packages. In a relatively small period of time the modern programmer can create, debug, and distribute large programming tasks. In the past, rapid development of such software required that a large group of programmers spend a substantial amount of time. With editing environments, such as WebGain's VisualCafe, a single programmer can construct and test a software package in a very efficient manner.

Current technologies are also designed to work well with the Internet. Components specifically designed to operate over the World Wide Web and to expedite data transfer are commonplace in modern computing environments.

3.3 Project Goals

3.3.1 Overview

The central goal of this project is to produce a graphical analysis software package for accessing and plotting nuclear reaction rates of astrophysical species found in the REACLIB reaction rate library. Certain criteria utilizing today's technology must be met. First, the software must be cross-platform so that users of any operating system will be able to execute the program. The program must also be deliverable over the Internet or local area network. Third, the software must be user-friendly, employing intuitive interfaces with standardized features and menus. The program must provide customizable plots with features that allow the user to explore not only the REACLIB library but also the data used to create the plot itself. Lastly, the program must provide a method by which one may compare the library's reaction rates in addition to ones added by user input.

3.3.2 The Graphical User Interface

A set of graphical user interfaces must be constructed in order to control and manipulate the production of the reaction rate plots. An interactive chart of the nuclides as utilized in an online posting of the Caughlan and Fowler 1988 database is an excellent choice for an efficient, intuitive interface to choose the isotopes of interest. By embellishing this chart with other central controls and options, we can build a primary interface that will provide most of the features we wish to bring to the nuclear astrophysicist. It is also important that the user will also be able to select various nuclear reaction types.

3.3.3 Plotting Options and Output

Unlike the two visualization implementations previously mentioned, the software will provide the user with various plotting options that may be chosen at runtime. The software will present options which reactions to plot, for log-log or log-lin plotting, major and minor gridlines, color or black and white plots, adjustable axis scales, and a legend complete with the reaction's string representation and reaction rate units [17]. The software must offer graphical output either by hardcopy or a retrievable file type.

3.3.4 Reaction Rate Analysis Features

Several important analysis features will also be included with the software. The user will have immediate access to the fitting parameters extracted from the REACLIB reaction rate library for the currently selected reactions. The total reaction rate as well as each resonant and non-resonant component will be available for plotting and analysis. By giving the user the capability to add their own reaction rate fitting parameters, the user will be able to compare newly-determined rates with current REACLIB entries. One last requirement for the software will be to make available the data points used in the actual plot so one may retrieve the data for reproduction or other uses.

Chapter 4

The REACLIB aLIVe! Interface

4.1 Introduction to the Interface

The REACLIB aLIVe! interface consists of several resizable windows. With the exception of the *Isotope Selection* window, all windows in aLIVe! possess a menubar with two standard menu items: *Options* and *Help*. The menu items listed for each *Help* menu are *Help Topics*, *Contact us*, and *About*. If the *Help Topics* menu item is chosen, a window containing instructions and tips applicable to the current window is opened. The menu items contained in the *Options* menu are dependent upon the parent window, or in Java, a *frame*.

4.2 The *Isotope Selection* Window

The initial graphical user interface of the REACLIB aLIVe! package is the *Isotope Selection* window (see Figure 4.1). This interface allows the user to select the isotope(s) and reaction type(s) of interest. The *Isotope Selection* window consists of three main parts: an interactive chart of the nuclides, the reaction type checkbox panel, and a button panel.

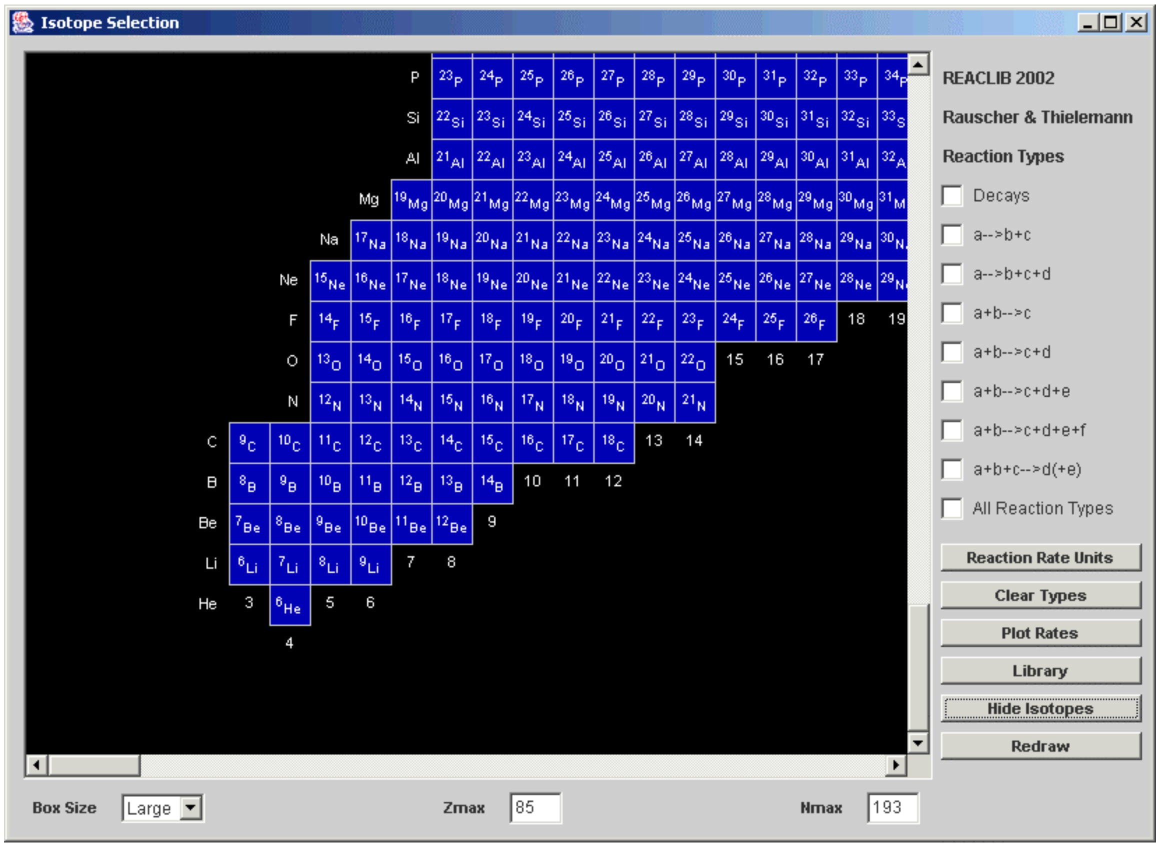


Figure 4. 1 The *Isotope Selection* window

4.2.1 An Interactive Chart of the Nuclides

The interactive chart of the nuclides represents isotopes currently available within the reaction rate library by mapping the proton number (Z) and the neutron number (N) to the vertical and horizontal axis, respectively. Each isotope is symbolized by a blue square. An isotope is selected for inspection when the user applies a *mouse-click* (left mouse button pressed) to the isotope's representative square, or *box*. The box undergoes a color change from blue to purple, which indicates that the isotope has been selected for investigation. The isotope is unselected by repeating this process.

Upon startup, the chart will by default only display isotopes up to a Z and N of 10 but may be extended to the maximum values given by the reaction rate library. The values are adjustable to a proton number of 85 and a neutron number of 193 by input text fields labeled "Zmax" and "Nmax". If the screen area covered by the chart is larger than what is viewable, scrollbars are automatically inserted in both horizontal and vertical directions. The user may also adjust the size of the boxes with the *Boxsize* dropdown menu to a *Small* or *Large* selection. To view a change to the extent of the chart or to the box size, the user presses the *Redraw* button in the right-hand side button panel of the *Isotope Selection* window.

4.2.2 The Reaction Type Checkbox Panel

As discussed previously, the rate library is categorized into eight reaction types depending upon the number of reactants and products within each nuclear reaction. A reaction type may be selected via a panel of checkboxes located at the right-hand side of the *Isotope Selection* window as seen in Figure 4.2. Also included is a checkbox labeled "All Types". When this option is checked, all other checkboxes are selected. While the *All Types* checkbox is selected, the user is unable to uncheck any other reaction type checkbox.

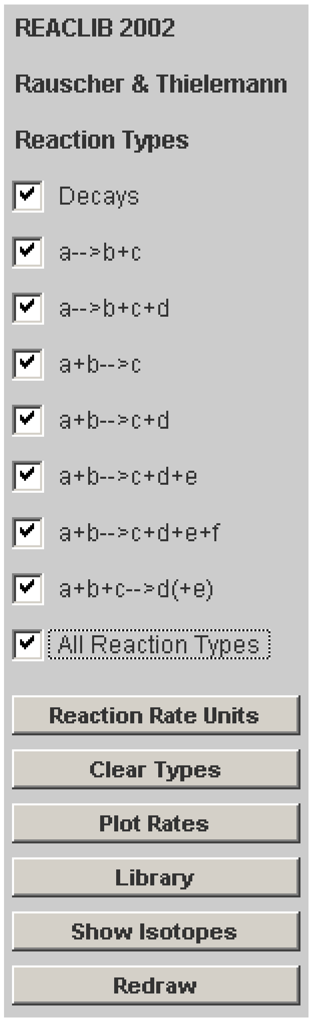


Figure 4. 2 The *Isotope Selection Button* panel

4.2.3 The Button Panel

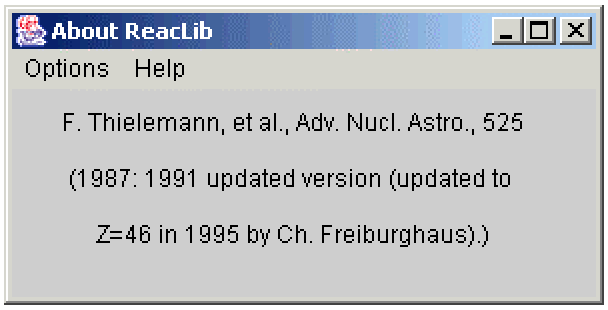
The button panel located at the right-hand side of the *Isotope Selection* window consists of six buttons labeled "Reaction Rate Units", "Clear Types", "Plot Rates", "Library", "Show Isotopes", and "Redraw". Pressing the *Reaction Rate Units* and *Library* buttons opens windows detailing the units of measure for each reaction rate and the full citation for the reaction rate library. The *Clear Types* button unselects all checked reaction type checkboxes. The *Plot Rates* button opens the *Plotting Parameters* window and initializes it with the chosen reactions. At least one isotope and one reaction type must be chosen before pressing the *Plot Rates* button. The button labeled "Show Isotopes (Hide Isotopes)" toggles on and off the isotope labeling within the chart of the nuclides. The box size option must be set to *Large* in order to view the labels. The *Redraw* button must be pressed to display any format changes to the interactive chart.

4.3 The *Library Citation* Window

The *Library Citation* window in Figure 4.3, which is opened by pressing the *Library* button of the *Isotope Selection* window, lists a full citation to the REACLIB reaction rate library. Here the *Options* menu contains *Print Window*, *Save as *.dat*, *Copy Text*, and *Close*. When the user chooses the *Save as *.dat* option, he/she will be prompted for a filename. An ASCII text file containing the full citation will be saved to the local directory (i.e., the directory containing the aLIVe! package).

4.4 The *Reaction Rate Units* Window

The *Reaction Rate Units* window in Figure 4.4, which is opened by pressing the *Reaction Rate Units* button of the *Isotope Selection* window, presents a table of reaction rate units vs. reaction type for the REACLIB reaction rate library. The *Options* menu for this feature allows the user to *Print Window*, *Save as *.dat*, *Copy Text*, and *Close*.

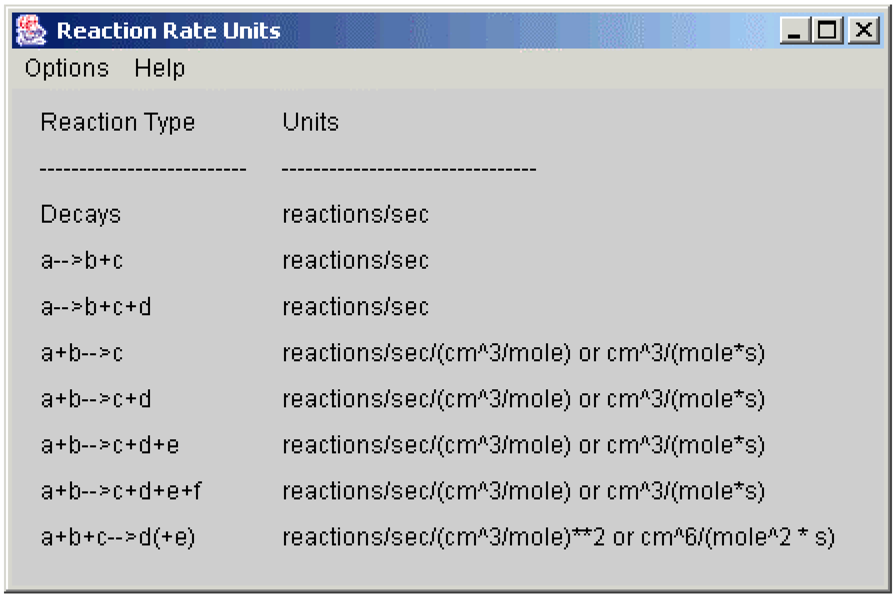


F. Thielemann, et al., Adv. Nucl. Astro., 525

(1987: 1991 updated version (updated to

Z=46 in 1995 by Ch. Freiburghaus).)

Figure 4. 3 The *Library Citation* window



Reaction Type	Units
Decays	reactions/sec
a-->b+c	reactions/sec
a-->b+c+d	reactions/sec
a+b-->c	reactions/sec/(cm^3/mole) or cm^3/(mole*s)
a+b-->c+d	reactions/sec/(cm^3/mole) or cm^3/(mole*s)
a+b-->c+d+e	reactions/sec/(cm^3/mole) or cm^3/(mole*s)
a+b-->c+d+e+f	reactions/sec/(cm^3/mole) or cm^3/(mole*s)
a+b+c-->d(+e)	reactions/sec/(cm^3/mole)**2 or cm^6/(mole^2 * s)

Figure 4. 4 The *Reaction Rate Units* window

4.5 The *Plotting Parameters* Window

The *Plotting Parameters* window, which is opened by pressing the *Choose Rates* button of the *Isotope Selection* window, consists of three main sections: a scrollable list of reactions chosen by the user, a set of plotting format options, and a button panel. Here, the user has the *Options* of *Print Window*, *Save as *.dat*, *Copy Text*, and *Close*.

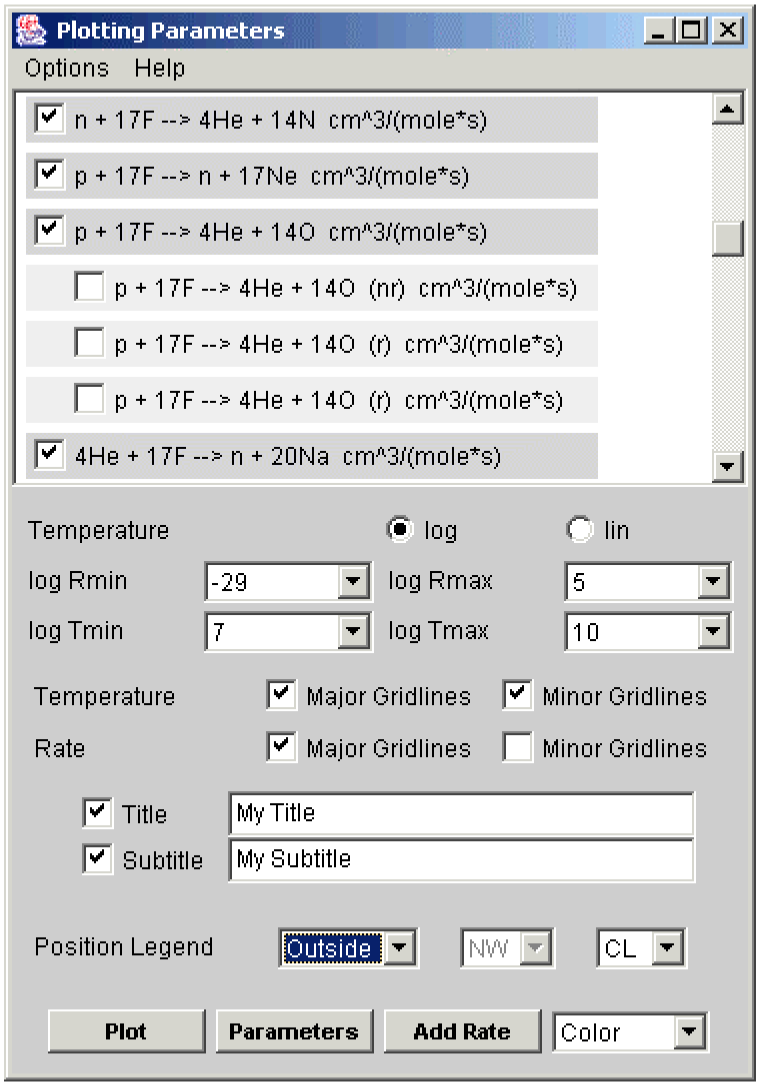
4.5.1 A Scrollable List of Reactions

The top portion of the *Plotting Parameters* window shown in Figure 4.5 consists of a scrollable reaction list. A checkbox and a label represent a reaction rate. Each is chosen by the user in the *Isotope Selection* window. A check next to a reaction label selects a reaction rate for plotting. Upon initialization, total reaction rates and single component reaction rates, which have a dark gray background color, are automatically selected. The light gray panels list resonant and non-resonant reaction rate components of the total rates. All reaction rate components must be selected by the user to be included in the plot.

4.5.2 A Set of Plotting Format Options

Below the reaction list is a set of formatting options for the plot. These include: *log* or *lin* temperature scaling for the horizontal axis (rate is always *log*), the range and domain, major and minor gridlines for each quantity, a title and subtitle, legend position, and a choice of color or b/w plotting. In this context, *log* refers to the logarithm base 10.

The reaction rates are valid over the temperature range 10^7 to 10^{10} Kelvin, therefore a linear or logarithmic scale may be chosen by the user. Since the reaction rates may vary by several magnitudes, this quantity is always plotted on a logarithmic scale. The range and domain of the plot may be specified through a set of dropdown menus. Each menu offers the user several choices for the minimum and maximum logarithmic values for reaction rate and temperature. The user is prompted if a maximum value is less than its minimum counterpart. Major and minor gridlines may be added to the horizontal and vertical axis of any plot. The user can also add a title or subtitle to the plot. Since the



[✓] n + 17F --> 4He + 14N cm^3/(mole*s)
[✓] p + 17F --> n + 17Ne cm^3/(mole*s)
[✓] p + 17F --> 4He + 14O cm^3/(mole*s)
[ ] p + 17F --> 4He + 14O (nr) cm^3/(mole*s)
[ ] p + 17F --> 4He + 14O (r) cm^3/(mole*s)
[ ] p + 17F --> 4He + 14O (r) cm^3/(mole*s)
[✓] 4He + 17F --> n + 20Na cm^3/(mole*s)

Temperature [✓ log] lin

log Rmin -29 log Rmax 5

log Tmin 7 log Tmax 10

Temperature [✓] Major Gridlines [✓] Minor Gridlines

Rate [✓] Major Gridlines [ ] Minor Gridlines

[✓] Title My Title

[✓] Subtitle My Subtitle

Position Legend Outside NW CL

Figure 4. 5 The *Plotting Parameters* window

legend could possibly obscure important line shapes and features, dropdown menus indicating a new legend position, are present. The positions are denoted by two sets: *inside* and *outside* of the plot's area. The *inside* positions listed are the N, S, E, W, NE, SE, NW, and SW corners of the plot's area. The *outside* positions are labeled NR, NL, CR, CL, SR, and SL, where R(L) indicates the right(left) side of the plot's area and N, C, or S represent the north, center, or south positions within that area. Finally, the user may choose color or black and white plotting with the last dropdown menu. The color or line style of each curve is automatically chosen from a preprogrammed list.

4.5.3 A Button Panel

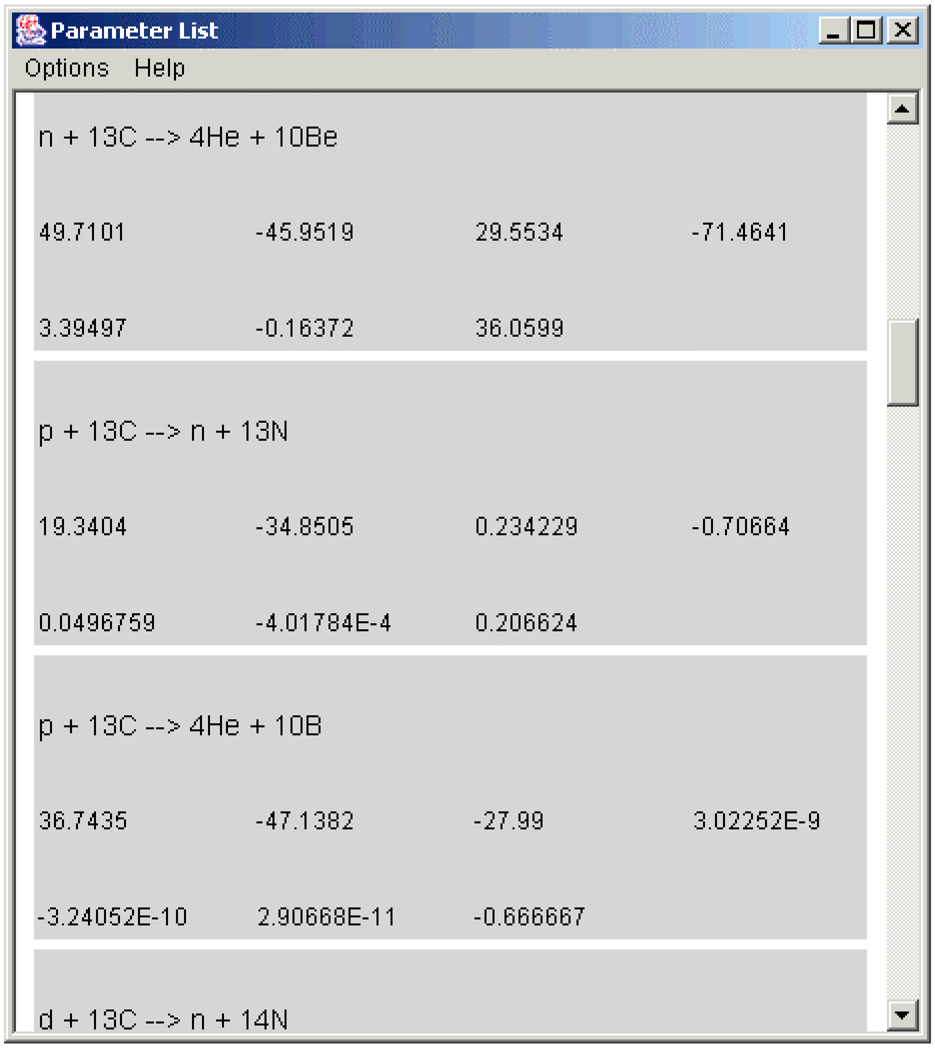
The button panel at the bottom of the *Plotting Parameters* window allows the user access to important actions. The *Plot* button opens the *Plot Display* window and also serves to refresh the current plot if the values of any formatting options have been changed. The *Parameters* button opens the *Parameter List* window. The *Add Rate* button begins the process in which a user is able to add a reaction rate to the current reaction list.

4.6 The *Parameter List* Window

The *Parameter List* window in Figure 4.6 displays the set of seven parameters for each reaction and component rate in the *Plotting Parameters* window's reaction list as a scrollable list of corresponding gray and light gray panels. The *Print Window*, *Save as *.dat*, *Copy Text*, and *Close Options* are also available from the menu bar.

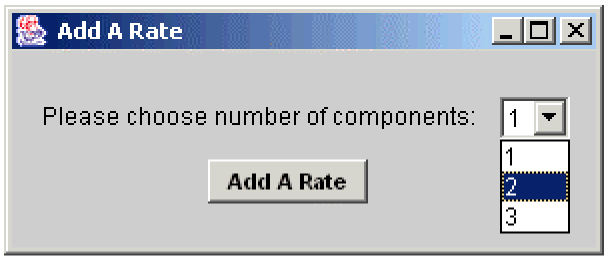
4.7 The *Add A Rate* windows

The *Add A Rate* function consists of the *Add A Rate* dialog box in Figure 4.7 and the *Add A Rate* parameter entry window shown in Figure 4.8. Upon initializing the *Add A Rate* function from the *Plotting Parameters* window, a dialog box appears that prompts the user to choose a one, two, or three component rate. After this selection, a custom window with the usual *Options* menu and containing text fields for the rate's title and each parameter is created. Upon pressing the *Add A Rate* button, the rate is added by title



```
n + 13C --> 4He + 10Be

49.7101        -45.9519       29.5534        -71.4641

3.39497        -0.16372       36.0599

p + 13C --> n + 13N

19.3404        -34.8505       0.234229       -0.70664

0.0496759      -4.01784E-4    0.206624

p + 13C --> 4He + 10B

36.7435        -47.1382       -27.99         3.02252E-9

-3.24052E-10   2.90668E-11    -0.666667

d + 13C --> n + 14N
```

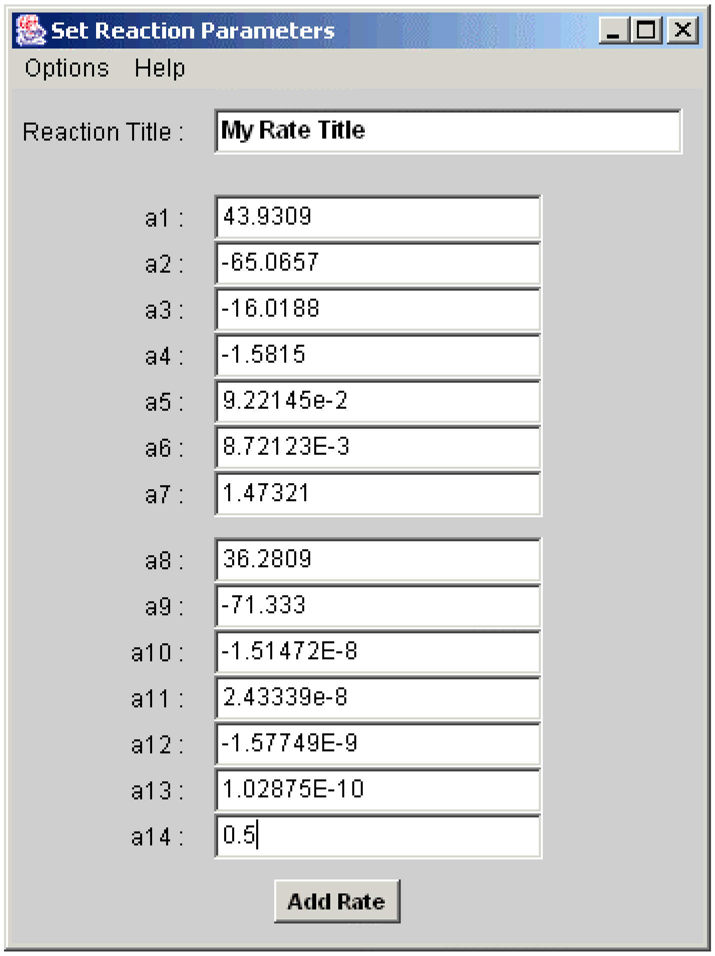
Figure 4. 6 The *Parameter List* window



Please choose number of components: 1

1
2
3

Figure 4. 7 The first window of the *Add A Rate* feature



Reaction Title : My Rate Title

a1 :	43.9309
a2 :	-65.0657
a3 :	-16.0188
a4 :	-1.5815
a5 :	9.22145e-2
a6 :	8.72123E-3
a7 :	1.47321
a8 :	36.2809
a9 :	-71.333
a10 :	-1.51472E-8
a11 :	2.43339e-8
a12 :	-1.57749E-9
a13 :	1.02875E-10
a14 :	0.5

Figure 4. 8 A window of the *Add A Rate* process

and possible components to the reaction list of the *Plotting Parameters* window. The parameters of the new rate are then available from the *Parameter List* window.

4.8 The *Plot* Window

The *Plot* window in Figure 4.9 consists of the plotting area and the legend. The plot's legend contains the line color or line type next to the corresponding reaction rate title. The *Options* menu is slightly different for the *Plot* window. The menu items are *Print*, *Save as *.ps*, *Rate vs. Temp Table*, and *Close*. The *Save as *.ps Option* will open a window prompting the user for a filename. A postscript file with this name will be saved in this program's local directory.

4.9 The *Reaction Rate vs. Temperature Table*

The user can view the numeric values that construct each curve per reaction rate by accessing the *Rate vs. Temp Table* menu option described in the previous section. The initial window shown in Figure 4.10 prompts the user to choose a reaction from a dropdown menu. By pressing the *Create Table* button, the user creates a custom, scrollable list of the data points for that curve as shown in Figure 4.11. The title of the resulting window displays the reaction of interest, and the units displayed match the reaction rate. Only reaction rates between 10^{-100} and 10^{100} units are shown. This window also has the usual *Options* menu.

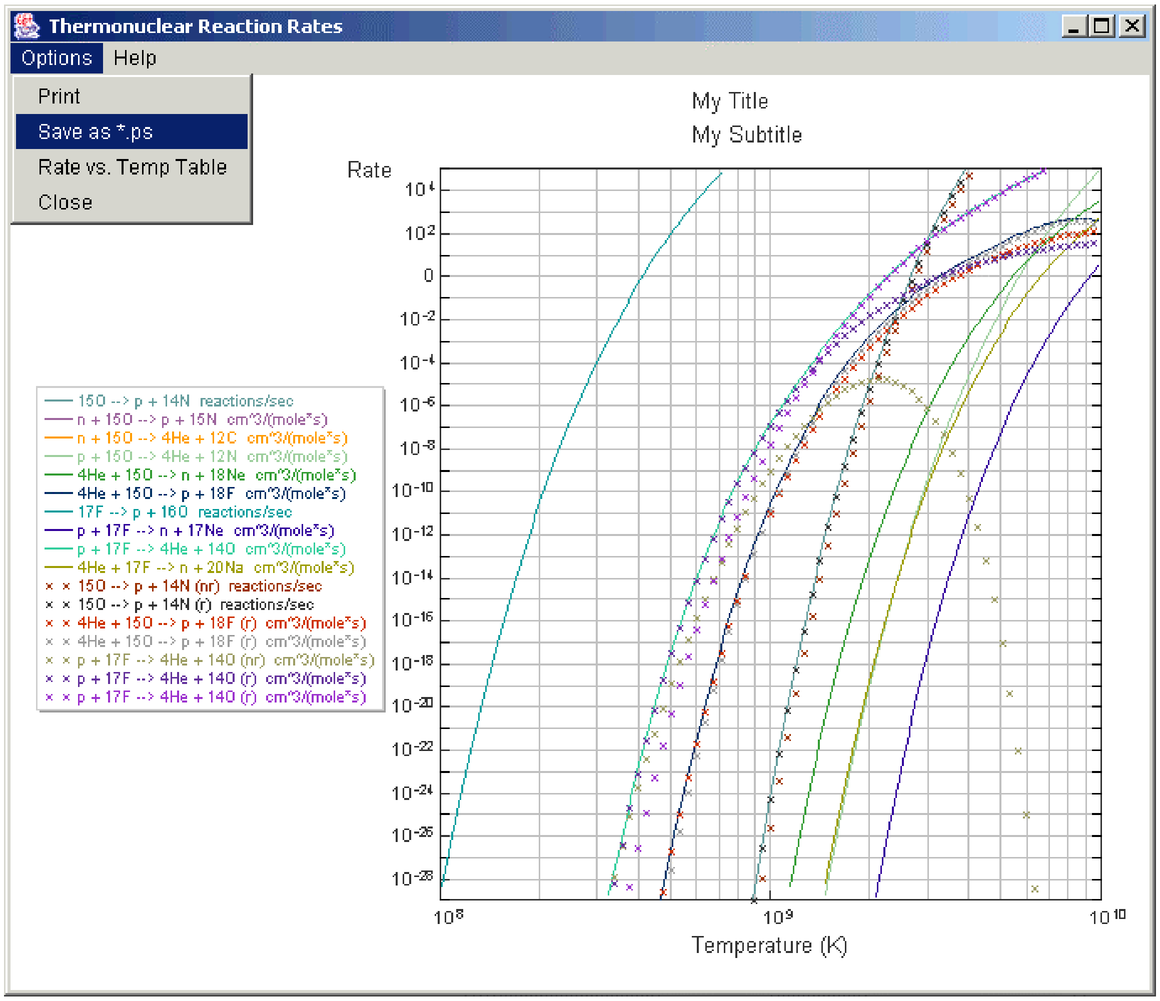
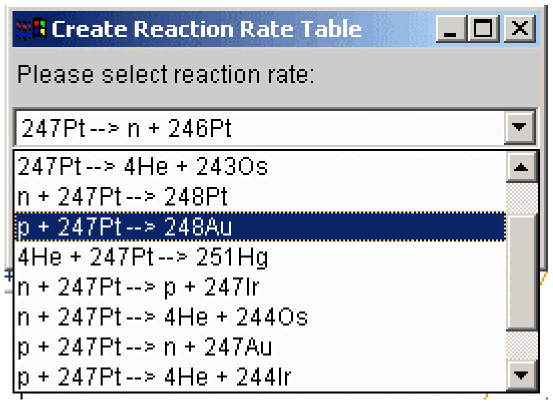

Thermonuclear Reaction Rates
Options
Help
Print
Save as *.ps
Rate vs. Temp Table
Close
My Title
My Subtitle
Rate
10^4
10^2
0
10^-2
10^-4
10^-6
10^-8
10^-10
10^-12
10^-14
10^-16
10^-18
10^-20
10^-22
10^-24
10^-26
10^-28
10^8
10^9
10^10
Temperature (K)
15O --> p + 14N reactions/sec
n + 15O --> p + 15N cm^3/(mole*s)
n + 15O --> 4He + 12C cm^3/(mole*s)
p + 15O --> 4He + 12N cm^3/(mole*s)
4He + 15O --> n + 18Ne cm^3/(mole*s)
4He + 15O --> p + 18F cm^3/(mole*s)
17F --> p + 16O reactions/sec
p + 17F --> n + 17Ne cm^3/(mole*s)
p + 17F --> 4He + 14O cm^3/(mole*s)
4He + 17F --> n + 20Na cm^3/(mole*s)
× × 15O --> p + 14N (nr) reactions/sec
× × 15O --> p + 14N (r) reactions/sec
× × 4He + 15O --> p + 18F (r) cm^3/(mole*s)
× × 4He + 15O --> p + 18F (r) cm^3/(mole*s)
× × p + 17F --> 4He + 14O (nr) cm^3/(mole*s)
× × p + 17F --> 4He + 14O (r) cm^3/(mole*s)
× × p + 17F --> 4He + 14O (r) cm^3/(mole*s)

Figure 4. 9 The *Plot* window

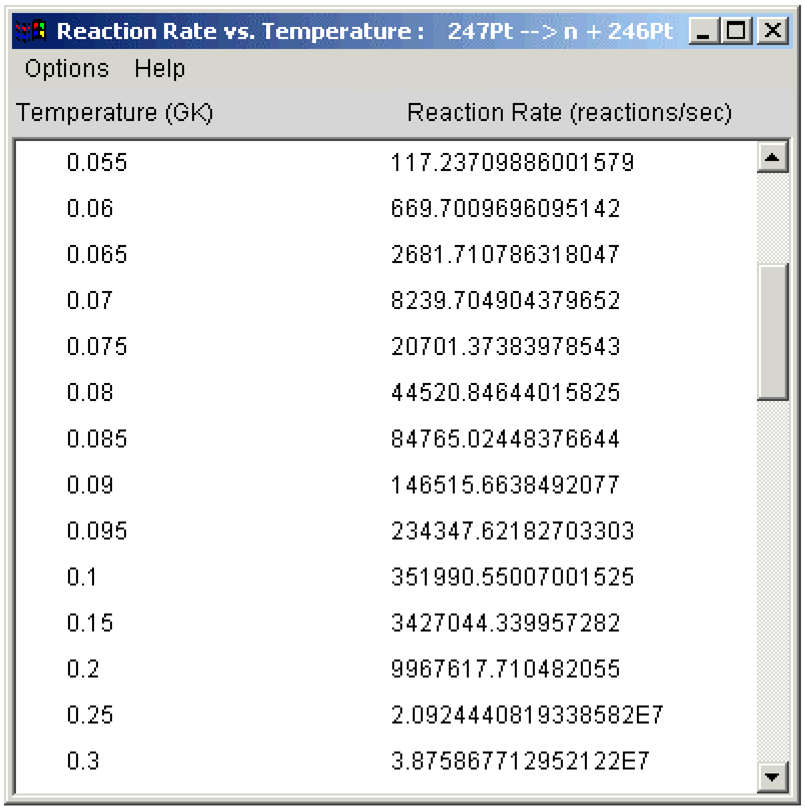


Please select reaction rate:

247Pt --> n + 246Pt

247Pt --> 4He + 243Os
n + 247Pt --> 248Pt
p + 247Pt --> 248Au
4He + 247Pt --> 251Hg
n + 247Pt --> p + 247Ir
n + 247Pt --> 4He + 244Os
p + 247Pt --> n + 247Au
p + 247Pt --> 4He + 244Ir

Figure 4. 10 Choose a reaction rate to create a table of data points



Temperature (GK)	Reaction Rate (reactions/sec)
0.055	117.23709886001579
0.06	669.7009696095142
0.065	2681.710786318047
0.07	8239.704904379652
0.075	20701.37383978543
0.08	44520.84644015825
0.085	84765.02448376644
0.09	146515.6638492077
0.095	234347.62182703303
0.1	351990.55007001525
0.15	3427044.339957282
0.2	9967617.710482055
0.25	2.0924440819338582E7
0.3	3.875867712952122E7

Figure 4. 11 The *Reaction Rate vs. Temperature* Table

Chapter 5

Special Programming Issues

5.1 What is *Java*?

5.1.1 The Object-Oriented Paradigm

Java and C++ belong to a set of programming languages called object-oriented languages. An object-oriented language is one that "adheres to the object-oriented development paradigm" [18]. The object-oriented paradigm is a natural, more understandable form of programming and is assumed to be cognitively similar to the way human beings perceive and understand the real world. This paradigm can be more strictly defined by several key concepts seen throughout Java.

All Java programs are composed of models known as *classes*. A class is essentially a template for any abstract or real world item. Classes include all of the attributes and functionality for an independent portion of the system. An *object* is an *instantiation* of the class (i.e., the object represents a specific state of the class). For example, the human being is a class, while "Travis Miller" is an object (an instance of the class human beings).

Methods are predefined operations within the class that provide functionality to the object. In other words, methods model an object's behavior. Methods accomplish this by manipulating an object's data-structures. The data-structures of an object can be declared *private*, or inaccessible. Access to private data can only be obtained by calling methods within that object.

Encapsulation is the process of keeping methods and data together, while allowing access to the object through only its methods. Therefore, the information defining an object's state is effectively hidden from other objects and is only read via methods.

Another key aspect of the object-oriented paradigm is *inheritance*. Inheritance allows the Java programmer to create a new class (*subclass*) from a higher-level class (*superclass*). The subclass not only contains all of the methods and data of the superclass, but also differs from the superclass by the addition of new data and methods. In Java, a subclass can have only one superclass, or the subclass *extends* the superclass. This is known as *single* inheritance. Other object-oriented languages, such as C++, may have *multiple* inheritance. Java can apply many features of multiple inheritance by implementing an *interface*. Interfaces and their implementation will be discussed later in this chapter.

The concept of *polymorphism* is also an essential part of the object-oriented paradigm. Polymorphism allows different objects to respond differently to the same stimuli. For example, a compact car and a sports car behave differently when the accelerator is pressed. A more complete discussion of the object-oriented paradigm and its application in Java can be found in the literature [19, 20].

5.1.2 Benefits of the Object-Oriented Paradigm

The object-oriented qualities of the Java language offer many innate advantages. One very apparent advantage is the natural and more understandable thought processes involved. The nature of Java allows the programmer to decompose a system into independent entities and map the problem at hand into classification hierarchies.

Another benefit of Java's structure is the efficient reuse of software. Reusable software leads to a higher productivity level and a solid foundation of well-proven classes on which to build.

Java's object-oriented nature also enhances communication between programmers, designers, and the client since all participants share common representations of the modules that compose a program.

5.1.3 The Java Programming Language

When the computing community discusses Java, they are actually referring to three distinct elements: the Java programming language, the Java Virtual Machine (JVM), and the Java platform [21]. When all of the above elements are used together, they are known as the Software Development Kit, or *SDK*. Several versions of the Java SDK are available for download from Sun Microsystems [22].

The Java programming language is the language and syntax used to write all forms of the Java class. Java has introduced many programming features that are not characteristic to other object-oriented languages like C++.

Unlike C++, Java is a *cross-platform* programming language. Java's cross-platform nature is "the most important promise of Java technology"[21]. That is, once a Java class has been written, it will run on any operating system that supports Java. For example, a Java program written on a Macintosh computer will execute on a Windows or Unix-based operating system. The Java's *portability* can now be seen in a number of devices, not just computers. In fact, Java can currently be installed on or integrated into cell phones and Personal Digital Assistants, or *PDAs* [23].

Java is safer and generally more reliable than other languages. For instance, the Java programming language will not automatically convert data types like C++. In most instances, one must explicitly convert data types [24]. By requiring manual data type conversions, Java ensures that the programmer's data structures are the intended ones.

A four-layer security process ensures Java's reliability. Java programs must pass through the Java compiler, the Java Virtual Machine, the class loader, and finally, API-specific security for applets before execution is allowed to take place [24]. By performing these types of error checks, the Java user will likely encounter a non-hostile, reliable programming process.

5.1.4 The Java Virtual Machine

In order for a computer to support Java, it must have the Java Virtual Machine (JVM) installed. Today, a JVM exists for nearly every operating system widely available. The JVM is necessary to run compiled Java programs, or *classfiles*, but how are the classfiles produced?

When a program written in the Java language is *compiled*, it is converted into *bytecode* by the Java *compiler*. Java bytecodes are a "special set of machine instructions that are not specific to any one processor or computer system" [25]. The JVM interprets the bytecode within the classfiles and runs the Java program. Therefore, Java is said to be an *interpreted* computing language. Interpreted languages exchange portability with computing speed [19]. Procedural languages, such as FORTRAN and C, run only on the compiling platform, but this limitation allows them to execute with high efficiency. New technologies, like Just-In-Time (JIT) compilers, speed up program execution by replacing sections of Java bytecode with native-platform machine language on the fly [21].

5.1.5 The Java Platform

The Java platform is a standardized set of classes that are available with every Java SDK installation. Java classes within this set are partitioned into similar groups called *packages*, and the Java platform is comprised of several packages, each with a central functionality. For example, packages exist for input/output, graphics, networking, etc. [21]. The evolution of the Java can be seen in the subsequent releases of larger and more functional Java platforms. Currently there are five major Java platforms freely available from Sun Microsystems [22]. They are denoted as Java 1.0 through Java 1.4 and are usually split into two groups. Java 1.0 and 1.1 are known as Java and all platforms released after Java 1.1 are called Java 2. A JVM built for the Java 2 platform can execute Java programs written with an earlier release of the Java platform, that is, the Java 2 platform is *backwards compatible* [20].

5.1.6 Javadocs: Automated Documentation

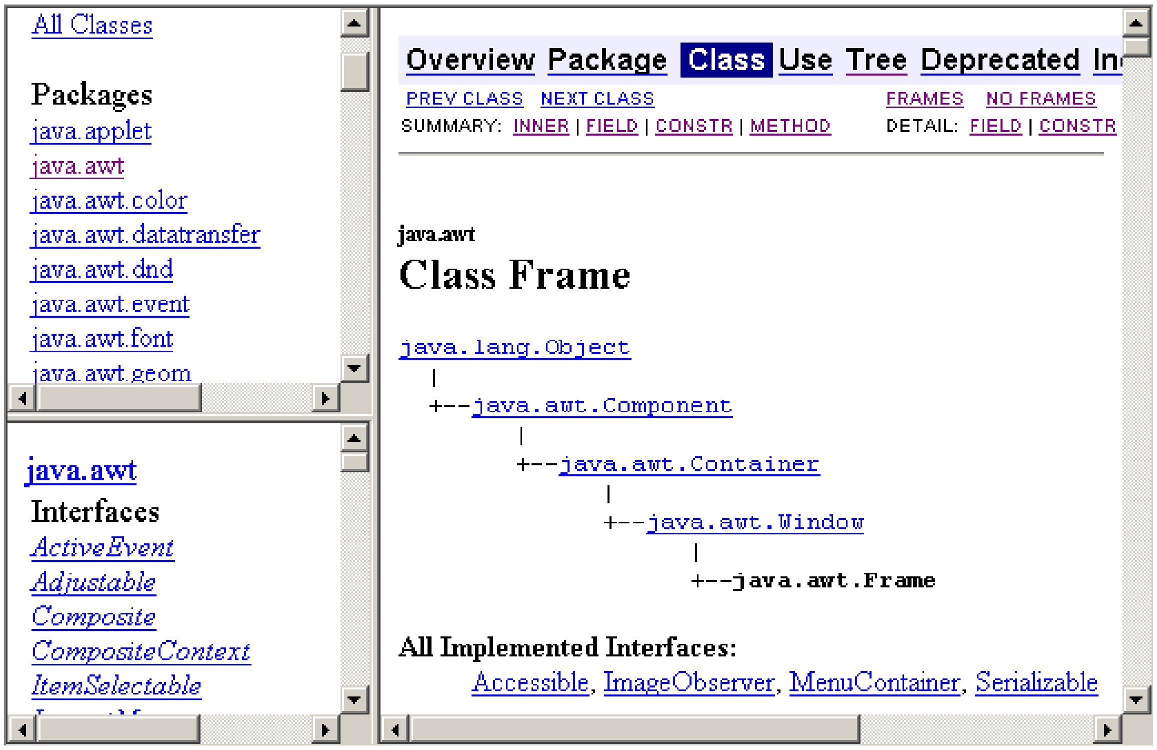
The Java SDK comes equipped with *javadoc*, a tool that generates industry-standardized documentation automatically [20]. The default output format is HTML and it can be appended to the Java platform documentation seamlessly. By entering the correct documentation tags in the Java source, the programmer can provide all relevant information needed to construct the files. The documentation may also include HTML tags used to customize the attributes of your text [20]. In Figure 5.1, multi-framed documentation for the *java.awt.frame* class is shown. From this interface, the user can quickly travel between documentation for any package or class available.

5.1.7 The Java Archive Utility

The *jar* utility included with the Java SDK can produce and modify Java Archive files, or *JAR*s [21]. A JAR file uses ZIP compression technology to package a set of Java classes and any related resource files. All JAR files must also include a manifest file that lists the JAR's contents and other relevant information.

5.1.8 Advantages of utilizing the Java 1.1 Specification

The Java platform utilized in the production of REACLIB aLIVe! is the Java 1.1 release. There are two characteristics of the Java 1.1 platform that are advantageous in the fabrication of this analysis tool: mass distribution of Java 1.1-enabled web browsers [26] and low memory consumption by the AWT User Interface components.


All Classes

Packages
java.applet
java.awt
java.awt.color
java.awt.datatransfer
java.awt.dnd
java.awt.event
java.awt.font
java.awt.geom

java.awt
Interfaces
ActiveEvent
Adjustable
Composite
CompositeContext
ItemSelectable

Overview Package Class Use Tree Deprecated In
PREV CLASS NEXT CLASS FRAMES NO FRAMES
SUMMARY: INNER | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR

java.awt
Class Frame

```
java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--java.awt.Window
                    |
                    +--java.awt.Frame
```

All Implemented Interfaces:
Accessible, ImageObserver, MenuContainer, Serializable

Figure 5.1 HTML documentation created by the *javadoc* tool

Java 1.1, which was released in March 1997 by Sun Microsystems, has been integrated into nearly every release of most mainstream web-browsers and operating systems [24]. Since Java is backwards compatible, this almost insures that the user will have the capability to run the aLIVe! software without downloading or improving the current state of the machine. The graphic user interface (GUI) package delivered with the Java 1.1 platform is known as the Abstract Windowing Toolkit, or AWT [25]. The AWT offers the Java programmer the ability to construct user interface components such as windows, buttons, dropdown menus, etc. The GUI components within the AWT conform to the operating system running the class files and take the appearance of the local GUI components [24]. For example, a Microsoft Windows 98 Java GUI will have the "windows" look, while the same program will appear differently on a Macintosh with OS 9 [27]. The functionality is the same across both operating systems.

With the release of the Java 2 platform, a new package called *javax.Swing* was made available to programming public. Swing GUI components have a standardized look across operating systems. So, a program with Swing GUI components will look identical on any operating platform [27]. Utilization of the Swing components may seem preferable over the use of AWT components, but unfortunately the Swing components require significantly more computer memory to display and run [28]. As we will see in a later section, memory consumption was a primary issue in the construction of the aLIVe! software.

5.2 Object Serialization

5.2.1 The Serializable Interface

As before mentioned, the Java programming language is an object-oriented language with *single* inheritance. To overcome this condition, a Java class may *implement* an *interface*. "An interface is a reference type closely related to the class" [21]. When a class implements an interface it inherits all of the *abstract* methods of that interface. Abstract methods are empty (i.e., they contain no functionality). All of the methods contained in the interface must be included within the class which implements it. Each method to be utilized by the class must contain Java code to define the functionality of the method. More information concerning Java interfaces may be found in the literature [21].

An example of an interface is as follows. Consider the MouseListener interface. It consists of the following abstract methods: mousePressed, mouseReleased, mouseEntered, mouseExited, and mouseClicked [27]. Now suppose we have written a class with AWT components and wish to add the mouse interaction. We would implement the MouseListener interface and add Java code to the methods, which would define actions fired by the user's mouse.

The *Serializable* interface is one of the most important features of the *java.io* package. Any instance of a class that implements the Serializable interface can be *serialized*. When an object is serialized, it is converted into stream of bytes that are read into a file. The state of the object is stored within a *filename.ser* file. The file may then be *deserialized* into a copy of the original object, which can be accessed during *Runtime* (i.e., during program execution).

5.2.2 Java Serialization Techniques

The Serializable interface is a special type of interface called a *marker* interface. A marker interface does not require any methods in order to be implemented. The two abstract methods utilized in this program are the writeObject and readObject methods, which take as arguments an ObjectOutputStream or an ObjectInputStream object, respectively [21]. With the above methods, the ObjectOutputStream and the ObjectInputStream define a path between the class and its serialized state. The writeObject method is invoked to write to the serialized file, and the readObject method is called to read from the file. The literature provides more discussion [29].

5.2.3 A Program to Parse REACLIB

Once an object is serialized, it can be deserialized and processed by the JVM quickly. It is for this reason that we have chosen object serialization as a way to rapidly give the user access to the data within the REACLIB Nuclear Rate Library. In order to do so, the data that constructs REACLIB must be grouped, or parsed, into "digestible" pieces. A Java class called *REACLIBParser* along with a program, *ParsedReaction*, has been written to accomplish this.

REACLIBParser and ParsedReaction contain code that reads REACLIB's data from a file. The text is partitioned by reaction and used to form a ParsedReaction object for each of the 5411 isotopes available. In the creation of each ParsedReaction object, the text is converted to several primitive data types, which can be read into Java for use in calculation. REACLIBParser sorts the ParsedReaction objects into a set of serializable *ReactionClass1* objects.

5.2.4 The ReactionClass1 Object

The ReactionClass1 object contains all of the information necessary for the aLIVe! software to operate on an isotope. The objects are grouped into sets by the proton number and neutron number of the heaviest reactant of the reaction. The groups are then serialized into a file. For example, all ReactionClass1 objects that have ^{13}C as the primary reactant will be serialized to a file called *iso6_7.ser*. When an isotope is chosen from the *Isotope Selection* window and the *ChooseRates* button is pressed, the corresponding serialized file is retrieved from the computer's hard disk or is transferred over a network. This allows instant access by the JVM to data that have been presorted for display or consumption by aLIVe!. Table 5.1 contains the information available for each ReactionClass1 object. This table lists the data type, variable name, and an explanation for each entry.

Table 5. 1 The ReactionClass1 data list

Data Type	Variable Name	Explanation
integer	reacIndex	The reaction type
integer	numberReactants	The number of reactants
integer	numberProducts	The number of products
String	reacString	The text representation of the reaction
String	refString	A reference to author and other information
boolean	ecFlag	True if electron capture
boolean	reverseR	True if a reverse reaction rate
boolean	resonant	True if resonant
boolean	nonResonant	True if nonResonant
Point Array	isoIn	An array of Z, N pairs representing the reactants
Point Array	isoOut	An array of Z, N pairs representing the products
double	Q	The Q-value
double	p0 through p6	The seven fitting parameters

ReactionClass1 also includes two methods: rate and serializeIt. The rate method takes a temperature in gigaKelvin as an argument and returns the reaction rate. The serializeIt method groups ReactionClass1 objects together by proton and neutron number and serializes them into a *isoZ_N.ser* file. Each serialized isotope file is quite small with an average file size of 1 to 3 kB each. The entire serialized library is only 14.2 MB and 7.8 MB when applying zip file compression.

The size of the original ASCII library file is 9.47 MB, which is slightly smaller than the serialized library. This is because the serialized file contains more than the original library. It encapsulates both the data and the class definitions for manipulating the data. REACLIB contains only the data, but contains little information about how to use the data to calculate observables. That must be supplied as separate information in order to use REACLIB. The Java objects contain both the REACLIB data and all rules (methods of the classes) for manipulating that data to obtain physical observables. Further, these are "Write once, play anywhere" objects meaning that they are cross-platform. Although REACLIB is "cross-platform" since it is an ASCII file, the standard codes that need REACLIB and construct physical quantities from it generally are not (e.g., typically F90).

5.3 Standalone Software

5.3.1 The Java Application

The REACLIB aLIVe! software can be distributed to the community in two forms: as a Java application and as a Java applet. A Java application is a set of Java classes that run on the user's computer and have complete access to the computer's *System* environment. Alternately, the Java applet is a set of Java classes that is downloaded within the context of the web browser and is executed only within the web browser. The Java applet has several limitations, which will be discussed in a later section but has the advantage that no installation (beyond that of a Java-capable web browser) is required on the client computer.

5.3.2 The *Options* menu

As an application, the aLIVe! software has the ability to print from the desktop, read and write to the disk, and run executables (i.e., make a call to the operating system) [18]. The *Options* menu of the aLIVe! window utilizes several of these abilities of the Java application. Note that only the *Isotope Selection* window does not contain a menubar. A general set of actions is used with each window except the *Plot Display* window. This set includes *Print Window*, *Save as *.dat*, and *Close*. The set of *Options* for the *Plot* window includes *Print Graph*, *Save as *.ps*, *Rate vs. Temp Table*, and *Close*.

5.3.3 The Ability to Copy and Paste Text

As discussed in Chapter 3, any text presented to the user can be copied to other applications. This capability is implemented by utilizing classes within the *java.awt.datatransfer* package, which operates on both *system* and *local* clipboards. A local clipboard runs only in the context of the JVM, while the system clipboard operates within the full operating system environment and has access to programs outside the JVM. The local clipboard allows a copy-paste between two Java programs, but to copy from a Java program and paste to a spreadsheet program like Excel not running under the JVM requires the system clipboard. The general "copy and paste" coding techniques seen in aLIVe! are adapted from publicly available websites, which are also open source [30].

5.4 Web Delivery

5.4.1 The Java Applet

Distribution of the aLIVe! software over the internet has been a primary goal of this project. The Java applet offers us the ability to dispense the aLIVe! software through a local network or over the World Wide Web. The applet is like an application in that it is a group of class files that work together as a software product. The class files are downloaded to the JVM via the web browser, and the downloaded applet must run within the context of the web browser. Since the Java applet must run within the context of a

Java-enabled web browser, it is restricted to the amount of memory set by the Java browser plug-in.

5.4.2 Memory Consumption

Since the Java applet must run within the context of the web browser, it is allowed access to the computer's memory (RAM) via the Java Plug-In. The amount of memory allotted at any one time to an applet is set through the Java Plug-In. In order to allow web access to the software by the widest range of users, memory usage must be kept to a minimum. The early Mac implementation of the applet required a large amount of memory. The large memory requirements for Mac implementation appear to be a memory management issue in the Mac version of the Netscape browser. By using the Mac's Applet Runner program, the applet version of the software appears to run properly with reasonable memory consumption. A study to examine which platforms and computing configurations that would run aLIVe! successfully is available in Appendix B.

5.4.3 Security Issues and Restrictions

Since the Java applet begins execution as soon as all class files have been downloaded, stringent security restrictions have been put in place by Java to protect the client system. The applet, unlike the application, cannot print locally without permission from the user. Also, standard applet permissions do not allow the applet to read or write to the local disk or make system calls. To overcome this problem, the user is instructed to place a set of permissions into the *java.policy* file allowing the aLIVe! applet to write to the client disk and copy text to the system clipboard. The java.policy file is a text file that determines the security restrictions for the user's Java plug-in and is located in the <java.home>\lib\security\ directory on the client's computer. By adding text commands, which are available at http://csep10.phys.utk.edu/RatePlotter/policyAppend.html, to the java.policy file, the user can allow the applet (and ONLY this applet) to have the full function of the application version.

5.4.4 The Java Console

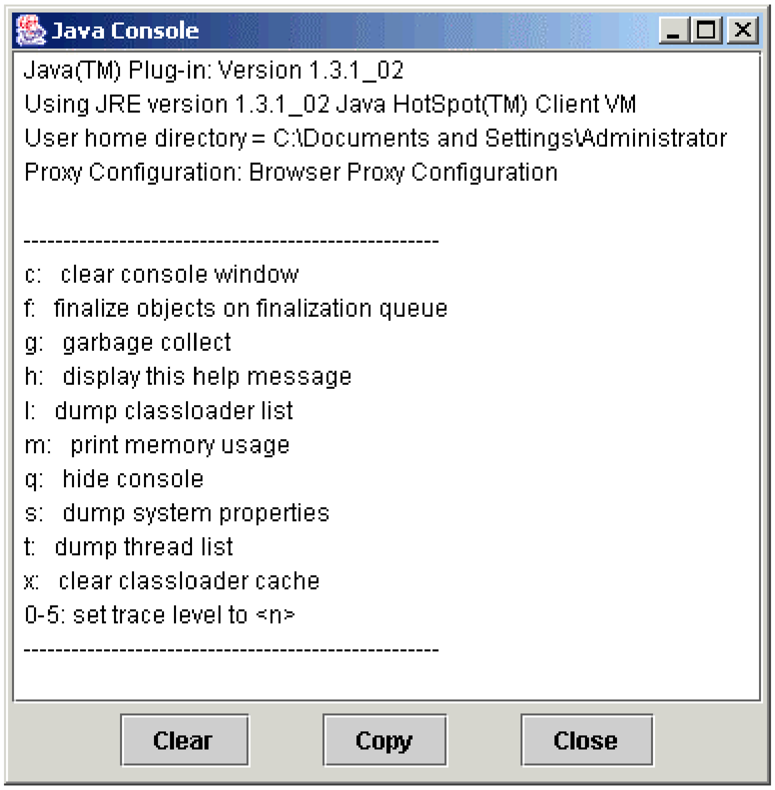
Since the applet is not allowed to write to the client-side disk, system output and error messages cannot be written directly to a system output window (i.e., a DOS window or Unix shell). To overcome this problem, the Java Plug-in comes equipped with the Java *console*. The Java console shown in Figure 5.2 is an active text output window for applets running through the web browser. The Java console has several features that allow the user to produce a system properties dump, view memory usage, and empty unused memory to increase capacity. Error messages for the user will print to the Java console in the applet version of the program.

5.5 Vector Graphics

5.5.1 Advantages of Vector Graphics.

The graphics output file format for the plots created with aLIVe! is a vector graphics format known as Postscript. There are several advantages to using vector-oriented graphical output versus a bitmapped output. A bitmap, or raster image, uses a grid populated with a fixed amount of colored pixels to produce an image much like a television [31]. Bitmaps are "the most common electronic medium for continuous-tone images, such as photographs or digital paintings, because they can represent subtle gradations of shades and color" [31]. Since a bitmap is constructed with a fixed number of pixels, a loss of resolution takes place if the image is scaled in any way. A larger or smaller grid size will force the screen to drop a few pixels, thus affecting the image clarity.

A vector graphics image is not resolution-dependent as with a bitmap [31]. A vector image is constructed by a "sequence of ...mathematical statements that place lines and shapes in a given two-dimensional or three-dimensional space" [32]. Instead of defining an area of pixels to display a line, for example, the vector representation would describe a series of points and then connect them. This can lead to a dramatic reduction in file size. Vector output is also edited with greater ease than its raster counterpart in many



```
Java(TM) Plug-in: Version 1.3.1_02
Using JRE version 1.3.1_02 Java HotSpot(TM) Client VM
User home directory = C:\Documents and Settings\Administrator
Proxy Configuration: Browser Proxy Configuration


----------------------------------------------------
c:   clear console window
f:   finalize objects on finalization queue
g:   garbage collect
h:   display this help message
l:   dump classloader list
m:   print memory usage
q:   hide console
s:   dump system properties
t:   dump thread list
x:   clear classloader cache
0-5: set trace level to <n>
----------------------------------------------------
```

Figure 5. 2 The Java console from Sun Microsystems

instances. Bitmaps must be modified pixel by pixel, while alterations to a vector image can be done through manipulation of the computerized vector objects.

5.5.2 Types of Vector Graphics Formats

The Postscript file format is only one type of vector image. Others, such as Encapsulated Postscript (EPS), Swiff (SWF), and Scalable Vector Graphics, or SVG (a form of XML), provide comparable capabilities to the postscript format, but also offer new and expanded features not previously available.

The EPS image format is a Postscript that is meant to be included within another document file [33]. The EPS file is a (relatively) standard format in all environments (i.e., it may be used to export and import Postscript language files across various computing platforms). An EPS file may contain text, graphics, images, and a bitmapped preview. The EPS preview provides a "crude representation" of the image if the system in question does not support the Postscript language [34].

Custom interactivity and the ability to rescale, rotate, and pan without resolution loss are the most important features produced in the development of the SVG and SWF formats. With SVG and SWF, the user can create custom responses to actions initiated by the mouse or the keyboard, which leads to innovative methods of communication and understanding between the user and the programmer. Both formats are very compact and are easily viewable over the World Wide Web [35].

5.5.3 Editing and Viewing Environments

Several interactive editing environments exist for all types of vector graphics file formats. For the Postscript file format, Adobe's Illustrator 10 and Macromedia's Freehand 10 offer the most versatile tools and graphic design capability currently available.

Macromedia's Flash MX software allows the user to graphically edit a Flash source file. By compiling, or "publishing", the Flash source file, the computer artist or programmer can create fully interactive and animated SWF files. The most important features of the Flash MX software are its highly intuitive interface and Actionscript, the programming language that produces custom interactivity. Actionscript is quickly approaching the versatility of a full object-oriented language complete with classes, methods, and objects [36].

SVG files are written in the Extensible Markup Language, or XML, and are easily implemented in any HTML website. The format is a web standard set forth by the World Wide Web Consortium, or W3C, in the SVG 1.0 specification [37]. Several editors are available for the SVG format, most notably Jasc's WebDraw and Adobe's GoLive [38, 39]. Both offer the user graphical editing of code and design, but no currently available tool for authoring SVG approaches the Flash MX tool (for authoring SWF) in power or versatility.

Viewing environments for all of the above formats are available. GSview, for example, displays the Postscript and EPS file formats and offers the user the capability to convert between the file types [40]. GSview is a visual interface for Ghostscript, which serves as an "interpreter" for the Postscript language [41]. The program is also an importer for Adobe's Portable Document Format, or PDF, with the additional ability to convert Postscript to this format [41].

The Flash Player and the Shockwave Player, available from Macromedia, are standalone and web plug-in viewers for the SWF format, respectively. The Flash player is also used by browsers for Flash content. The Shockwave player plays both Flash and output from Macromedia's Director software. Viewers for SVG include Adobe's SVG Viewer and Apache's Batik SVG browser, which operates on the Java 2 Platform [42, 43].

5.5.4 Conversion from Postscript to SWF

A plot created with aLIVe! and saved as a Postscript file can be converted into the SWF format by utilizing features available through the Adobe Illustrator and Macromedia Flash environments. By opening the plot in Illustrator, the user can export the file as Encapsulated Postscript [44]. The EPS file may be imported into Flash and used in conjunction with other Flash movies to create web-deliverable graphics and presentations with high visual quality and interactive control [36]. The EPS file is also editable within the Flash context. All portions of the graph including curves, labels, and gridlines may be deleted or scaled. Color and line style properties are also editable. Figure 5.3 demonstrates editing of an aLIVe! plot.

We have efforts underway to facilitate a direct path from Java to the SWF and SVG formats. Since SWF is an open source standard, one has the ability to investigate the structure of the format and thus create a set of Java classes to map graphics output to SWF. A third-party API from Flagstone Software provides this ability. Unfortunately, the API requires a substantial licensing fee for its use in a product but can freely be used for developmental purposes. Given this limitation, we have outlined a technique to export an XML document directly from Java, which can be imported into a custom SWF viewer. A number of Flash ActionScript commands exist that deal directly with XML. For SVG, students within our group have created a Java interface that allows the user to create simple shapes and text on a canvas and export the drawing directly to an SVG file. This prototype demonstrates the ability to export a graphics context from Java to SVG.

5.6 Distribution to the Scientific Community

5.6.1 A Java Applet over the World Wide Web

The aLIVe! software can currently be viewed over the World Wide Web with any Java equipped browser at http://csep10.phys.utk.edu/RatePlotter/start.html. The website includes instructions, system requirements, and screenshots of the software. Figure 5.4 shows the aLIVe! logo, which is included at the website, and was produced with Electric

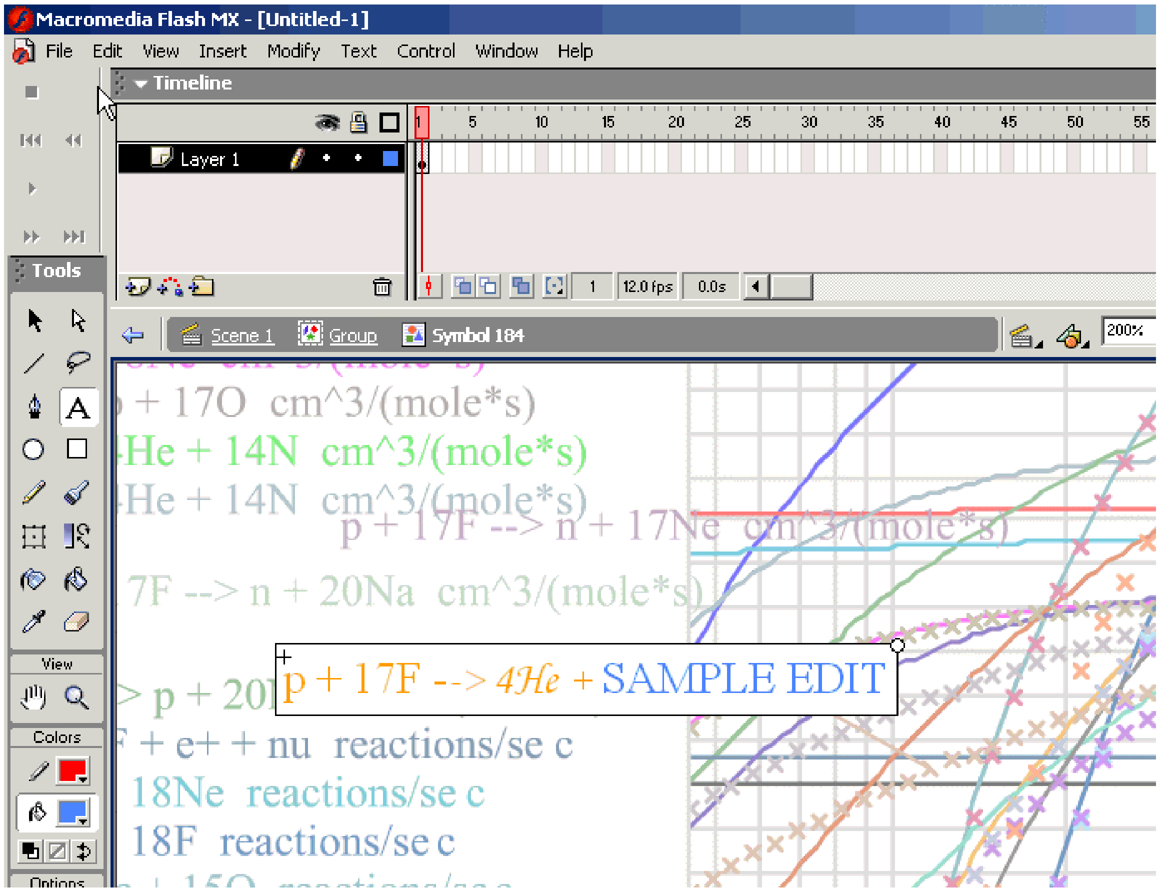


Figure 5.3 A reaction rate plot edited in the Flash MX environment

Figure 5.4 The REACLIB aLIVe! logo

Rain’s Swift3D. A Java button at the bottom of the web page launches the software. The website is also linked from http://www.nucatrodata.org [45].

5.6.2 A Compressed Java Application with Data Set

The standalone Java application can be downloaded as a Java Archive, or *JAR*, file at http://csep10.phys.utk.edu/RatePlotter/download.html. Here the user can also download a compressed ZIP file containing the serialized data files. The JAR file can be expanded at the command line by using the jar utility accompanying any Java Software Development Kit, or SDK, and the zipped data can be unzipped by WinZIP or any decompression utility supporting the ZIP file format. The “data” directory must be in the same folder as the aLIVe! class files.

Chapter 6

Conclusion

6.1 Summary

Nucleosynthesis calculations let researchers investigate where all matter in the universe originated. By solving sets of stiff, ordinary differential equations numerically, nuclear astrophysicists expand our knowledge of astrophysical phenomena such as the Big Bang, stellar evolution, and core-collapse supernovae. These simulations require, as input, the rate at which individual nuclear reactions occur. Large libraries of reaction rate data, such as REACLIB, that are constructed of theoretical and experimental rates are used in this process. Currently, technologies exist for viewing these libraries, but they are cumbersome and lack programming features that would assist nuclear astrophysicists.

We have developed a cross-platform, lightweight Java software package for plotting and analyzing the nuclear reaction rate library, REACLIB. The software package, REACLIB aLIVe!, is available to the scientific community as a standalone implementation or as a Java applet usable via the World Wide Web without local installation. The utilities available with the aLIVe! software package will allow scientists to compare current REACLIB rates with newly adjusted rates, create custom plots for papers and presentations, and provide new researchers a hands-on educational tool on the subject of nuclear reaction rates and nucleosynthesis. This package is the first efficient, interactive visualization of parameterized nuclear reaction rates of importance in astrophysics.

6.2 Future Enhancements

Several future enhancements of the software are being considered. A graphical representation of the reaction types as interactive vectors in the NZ-plane could give a more intuitive means of selecting reaction rates for viewing. Another enhancement would give the user the ability to plot ratios of multiple rates versus a predefined rate or plotting the ratio of any two chosen rates. The ability to save your rate into the library and the ability to directly export graphs and animated sequences into the SWF or SVG formats would be useful additions. Also a mouse-click response over a data point returning an ordered pair (temperature, rate) would prove useful.

A Java 2 version of aLIVe! utilizing the Swing components is another possible upgrade to the software. Swing components are more memory and processor intensive than the current AWT components and are less uniformly supported cross-platform in applets. However, these issues are largely irrelevant for the newest generation of desktop computers and a Swing version would give a more pleasing and uniform look to the graphical interface.

Bibliography

Bibliography

[1] Burbridge, E. M. Burbridge, G. R. Fowler, A. A. Hoyle, F. 1957, Rev. Mod. Phys., 29, 54.

[2] http://quasar.physik.unibas.ch/tommy/astro/reaclib/reaclib.nosmo.

[3] “Nuclear Masses and Binding Energy.” <http://library.thinkquest.org/3471/mass_binding_body.html?tqskip1=1&tqtime=1016>.

[4] Carroll, Bradley W. and Dale A. Ostlie. An Introduction to Modern Astrophysics. Massachusetts: Addison-Wesley, 1996.

[5] Hansen, C. J. and S. D. Kawaler. Stellar Interiors: Physical Principles, Structure, and Evolution. New York: Springer-Verlag, 1994.

[6] Clayton, Donald S. Principles of Stellar Evolution and Nucleosynthesis. Illinois: University of Chicago P., 1983.

[7] Online Journey Through Astronomy. Lightcone Interactive, LLC. Brooks-Cole, 1999.

[8] Hix, W. R. and F-K. Thielemann. “Computational Methods for Nucleosynthesis and Nuclear Energy Generation”. ApJ., 29 June 1999.

[9] F. Thielemann, et al., Adv. Nucl. Astro., 525 (1987).

[10] http://quasar.physik.unibas.ch/tommy/astro/reaclib/readme.

[11] Rauscher, T. and Thielemann, F-K. “Astrophysical Reaction Rates From Statistical Model Calculations”, Atomic Data Nuclear Data Tables **75** (2000) 1.

[12] Arnett, David. Supernovae and Nucleosynthesis. New Jersey: Princeton University P., 1996.

[13] Parete-Koon, Suzanne. "Reaction Rate of $^{17}F(p,\gamma)^{18}Ne$ and Its Implications for Nova Nucleosynthesis." Thesis U. of TN, 2001.

[14] Dessieux, Luc. "2P or Not 2P: The Reaction Rate $^{14}O(\alpha,2p)^{16}O$ and Its Implication on Novae and X-ray Bursts." Thesis U. of TN, 2002.

[15] “Thermonuclear Reaction Rates.” <http://www.phy.ornl.gov/astrophysics/data/cf88/index.html>.

[16] “Astrophysical Cross Sections and Reaction Rates (T. Rauscher).” <http://quasar.physik.unibas.ch/~tommy/reaclib.html>.

[17] “Table of Nuclides.” <http://www2.bnl.gov/ton/main.html>.

[18] Wright, Chris. Java. 2nd ed. Illinois: NTC/Contempary, 2000.

[19] Schildt, Herbert. Java 2: A Beginner’s Guide. California: Osborne/McGraw-Hill, 2001.

[20] Deitel, H.M. and P.J. Deitel. Java How To Program. 3rd ed. New Jersey: Prentice-Hall, 1999.

[21] Flanagan, David. Java in a Nutshell A Quick Desktop Reference. 3rd ed. California: O’Reilly and Associates: 1999.

[22] "Archive: Java(TM) Technology Products Download."
<http://java.sun.com/products/archive>.

[23] "JavaMobiles.com-list of JVM for PDA." <http://www.javamobiles.com/jvm.html>.

[24] Jaworski, Jamie. Java 1.1 Developer's Guide. 2nd ed. Indiana: Sams.net, 2000.

[25] Lemay, Laura and Charles L. Perkins. Teach Yourself Java 1.1 in 21 Days. 2nd ed. Indiana: Sams.net, 2000.

[26] "JAVA 1.1 compatible browsers."
<http"//www.informatik.uni-rostock.de/~gdb/isgci12/browsers>.

[27] Flanagan, David. Java Foundation Classes in a Nutshell. California: O'Reilly and Associates, 1999.

[28] "AWT vs Swing." <http://community.borland.com/article/0,1410,26970,00.html>.

[29] Flanagan, David. Java Examples in a Nutshell. 2nd ed. California: O'Reilly and Associates, 2000.

[30] "Java Tip 61: Cut, copy, and paste in Java."
<http://www.javaworld.com/javatips/jw-javatip61_p.html>.

[31] "About Bitmap Images and Vector Graphics."
<http://cit.asma.ru/Photoshop/c02im2.htm>.

[32] “vector graphics - a search WebServices definition - see also: vectorized.” <http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci213284,00.html>.

[33] “TechPubs Help Pages: Creating Encapsulated Postscript (EPS) Files.” <http://www.slac.stanford.edu/grp/techpubs/help/figures/eps.html>.

[34] “What is Encapsulated Postscript." <http://www-h.eng.cam.ac.uk/help/tpl/graphics/postscript.html>.

[35] “Introduction to SWF.” <http://www.openswf.org/spec/SWFfileformat.html>.

[36] Reinhardt, Robert and Snow Dowd. Flash MX Bible. New York: Wiley, 2002.

[37] “Scalable Vector Graphics.” <http://www.w3.org/Graphics/SVG/Overview.htm8>.

[38] “Jasc WebDraw.” <http://www.jasc.com/products/webdraw/>.

[39] “SVG Zone - 3rd party tools.” <http://www.adobe.com/svg/tools/other.html>.

[40] “GSview.” <http://www.cs.wisc.edu/~ghost/gsview/index.htm>.

[41] “Ghostscript Frequently Asked Questions.” <http://www.cs.wisc.edu/~ghost/doc/faq.html>.

[42] “Scalable Vector Graphics.” <http://www.adobe.com/svg/viewer/install/mainframed.html>.

[43] “Batik SVG Toolkit.” <http://xml.apache.org/batik/index.html>.

[44] Adobe Systems Incorporated. Adobe Illustrator 9.0 User Guide for Windows and Macintosh. California: Adobe Systems Inc., 2000.

[45] “Nuclear Astrophysics Data Sets.” <http://www.nucastrodata.org/data_list.html>.

Appendix

Appendix A: List of the aLIVe! Classes

AddRateDialog - Creates a dialog box prompting the user for the number of rate components he/she wishes to add.

AddRateFrame - Creates the *Add A Rate* window.

HorString - Draws a string horizontally to any Graphics object.

IsotopePad - Creates the interactive chart of the nuclides in the *Isotope Selection* window.

LibRef - Creates the *Library Citation* window.

myColors - A set of standard colors used by various classes.

ParamList - Creates the list of parameters for all selected reactions.

ParamListFrame - Creates the *Parameter List* window, which encases the ParamList class.

PixelConsumer - Along with the PSGr class, PixelConsumer allows Graphics output to a PostScript file.

PlotParams - Creates the *Plotting Parameters* window by encasing the PlotReactionList class. Also initializes the *Plot Display* window.

PlotReactionList - The interactive list of reaction rates chosen in the *Isotope Selection* window.

PSGr - Along with the PixelConsumer, this class allows Graphics output to a PostScript file.

RateGraphicsPad - Calls the PlotIt method from the StaticPlotter class to paint the plot on a canvas.

RatePlotFrame - Creates the *Plot Display* window by encasing the RateGraphicsPad class.

RatePlotter - The class that holds the main method for aLIVe!.

ReactionClass1 - The serializable class that holds all information for each isotope in REACLib.

SegreFrame - Creates the IsotopeSelection window by encasing the IsotopePad class.

StaticPlotter - The StaticPlotter class holds the PlotIt method that writes the plot to the RateGraphicsPad Graphics context.

TempRate - The listing of reaction rate versus temperature values.

TempRateDialog - The user can select a set of reaction rate versus temperature data points from this dialog box.

TempRateFrame - Creates the *Reaction Rate vs. Temperature Table* by encasing TempRate.

Units - The listing of all reaction rate units arranged by reaction type.

UnitsFrame - Creates the Reaction Rate Units window by encasing the Units class.

VertString - Draws a string vertically to any Graphics object.

Appendix B: System Requirements and Configurations

Java 1.1 (or higher) Runtime Environment (Java Plug-In) from Sun Microsystems with the maximum heap size of 96Mb (default value for the Java Runtime Environment).

Note: The plotter may not execute properly through Internet Explorer if the user has installed the Microsoft VM. On most windows machines simply installing the Java plug-in from Sun Microsystems will alleviate this problem.

The plotter has been tested to work properly on most system configurations. Below is a list of environments known to support the plotter:

Netscape 6.x or later using Microsoft Windows 98, NT, 2000, ME, XP with at least 128 Mb of RAM.

Microsoft Internet Explorer 5.x or later using Microsoft Windows 98, NT, 2000, ME, XP with at least 128 Mb of RAM.

Internet Explorer 6.x or later using Red Hat Linux 7.3 with at least 128 Mb of RAM.

Note: Since the *java.awt* package (Abstract Windowing Toolkit) creates windows with properties that are dependent upon the operating system, some users may need to resize each window to display all components correctly.

Netscape 6.x or later on Mac OS X on a 400MHz G4 with 768MB RAM

Note: All menubars featured in the screenshots below will be found in the main toolbar of your Mac. The "Options" and "Help" menus will change according to the window in focus.

Due to the amount of memory necessary for this applet to execute correctly, some computing configurations may not be able load the plotter properly. Below is a list of environments known not to support the plotter.

Netscape 6.x or later on Mac OS X on a 400MHz G3 with 384MB RAM.
Any Mac with OS 9

If any problems are encountered using the applet version of the software on a Mac, users are advised to run the applet through the Mac Applet Runner.

Vita

Eric J. Lingerfelt was born August 24, 1975 in Erwin, TN. After graduating Unicoi County High School with honors, he attended East Tennessee State University in Johnson City, TN where he received a Bachelor of Science with majors in Mathematics and Physics and the Outstanding Graduating Senior in Physics award. While at ETSU, he performed spectroscopic measurements of Mira variables at the Southeastern Association for Research in Astronomy (SARA) 0.9-meter telescope at Kitt Peak National Observatory with subsequent publication and poster presentation at the American Astronomical Society 191st Meeting in Washington, DC. In 1998 he was granted a teaching assistantship and the Science Alliance Fellowship from the Department of Physics and Astronomy at the University of Tennessee in Knoxville. In 2001 he received the Robert C. Lide Citation for Outstanding Laboratory Development for his work on a set of ten virtual astronomy laboratories. Also in 2001 he was awarded a full research assistantship from the Tennessee Educational Technology Initiative. The work within this thesis along with other astrophysics visualization initiatives was presented at the Conference on Computational Physics 2002 in San Diego, CA.