A Utility Program for Treating Composite Particles with the GRACE System

Masato Jimbo

Computer Science Laboratory, Tokyo Management College,
625-1 Futamata Ichikawa, Chiba 272, JAPAN
(e-mail: jimbo@tmc-ipd.ac.jp)

Abstract

A utility program has been constructed by the author to treat the processes including composite particles with the `GRACE` system which has been developed for the automatic computation of the matrix elements for the processes of the standard model and the minimal SUSY standard model (MSSM).

keywords: Composite particles; `UNIX`; `C`; `GRACE`

1 Introduction

For the simulations of the experiments in high-energy physics [1–3], we have to calculate the cross-sections for the processes with the final 3-body or more. We have already known within the standard model that the calculation of the helicity amplitudes is more advantageous to such a case than that of the traces for the gamma matrices with `REDUCE` [4, 5]. The program package `CHANEL` [6] is one of the utilities for the numerical calculation of the helicity amplitudes.

It, however, is hard work to construct a program with many subroutine-calls of `CHANEL` by hand. Thus we need a more convenient way to carry out such a work. Several groups have started independently to develop computer systems which automate the perturbative calculation in the standard model with different methods [7–11]. The `GRACE`

system [7], which automatically generates the source code for CHANEL, is one of the solutions. The system also includes the interface and the library of CHANEL, and the program package BASES/SPRING v5.1 [12] for multi-dimensional integrations and event-generations.

In recent works [13–19], we have developed an algorithm to treat Majorana fermions, which appear in the supersymmetric (SUSY) model [20], in CHANEL. In the standard model, we already have such particles as Dirac fermions, gauge bosons and scalar bosons in the GRACE system. There, however, exists another problem on fermion-number violating interactions. We have also developed an algorithm for this problem, and have constructed an automatic system for the computation of the SUSY processes by the algorithms above in the GRACE system [21].

Thus far the GRACE system works well for processes including elementary particles, such as those in electron-positron collisions. We, however, have to treat composite particles, *e.g.* proton, due to requests from experimentalists. Since a proton consists of several quarks, anti-quarks and a gluon, we have to handle multi-processes in the level of an elementary-particle process. In this paper, we discuss how to construct a utility program for such processes.

2 Outline of the GRACE system

In Fig. 1, we present the flow diagram of the GRACE system [22]. The GRACE system has become more flexible for the extension in the new version [23], and includes a new graph-generation package called '`grc`' [24], which is written by C. With this package, any graphs based on a user-defined model can be generated at any orders. The Feynman diagrams are drawn by the program package '`gracefig`' [25] in the new GRACE. It is necessary for us to make the interface and the library of CHANEL and the model file when we include new particles in the system.

The usage of the system is as follows:

1. Specify a physical process in an input file (the default name is '`in.prc`'; see Appendix A.1).
2. Execute '`grc`' to get an output file '`out.grf`' which consists of the definitions of the generated graphs.
3. Execute '`gracefig`' to view the generated graphs.
4. Execute '`grcfort`' to get source files written in FORTRAN, which consists of the subroutine-calls of CHANEL to calculate helicity amplitudes numerically as a complex number, parameter files and a '`Makefile`'.

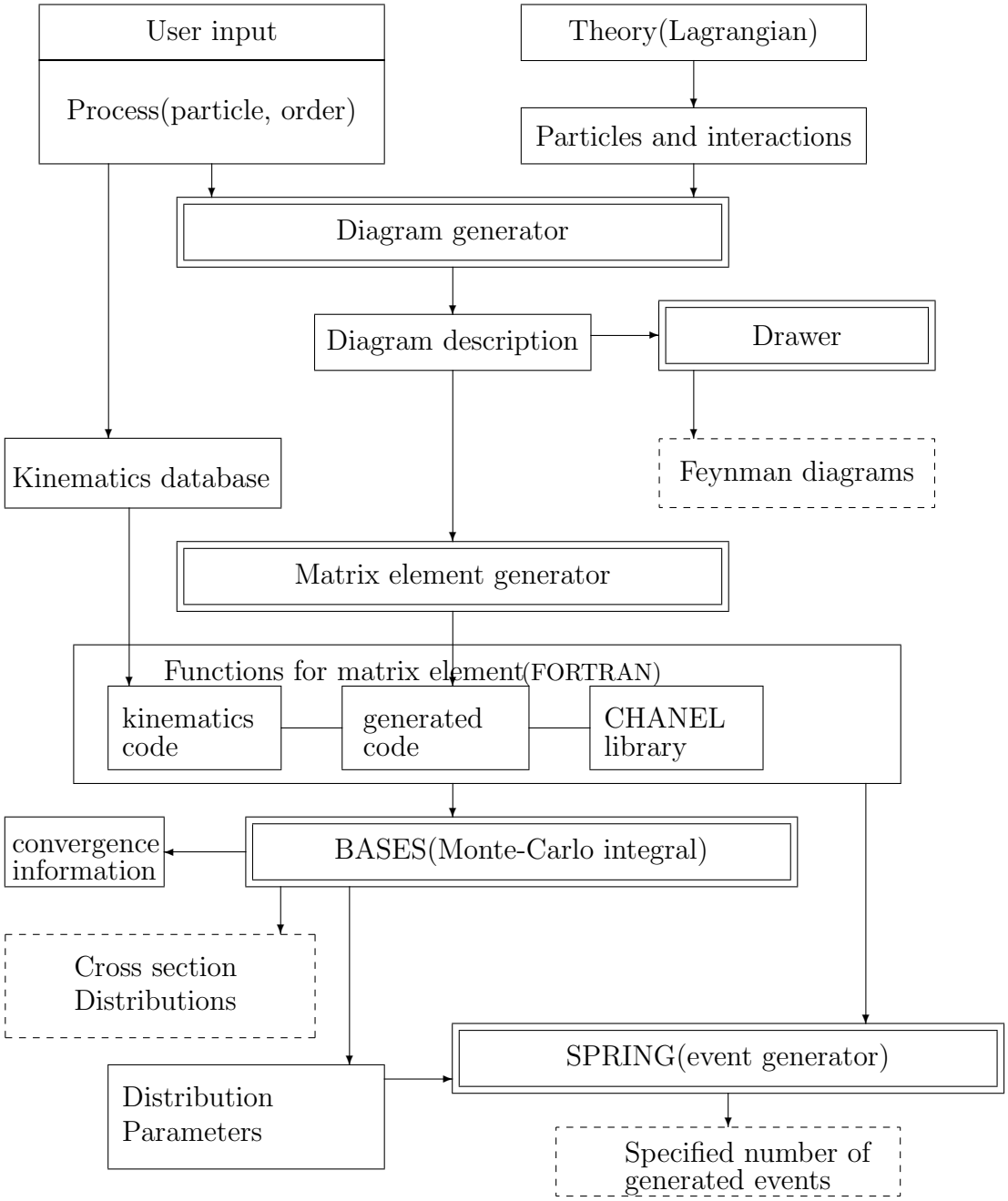

Fig. 1. GRACE system flow (after version 1.1)

5. Execute '`make`' command to get executable files '`gauge`', '`integ`' and '`spring`'.

6. Execute '`gauge`' to test the gauge invariance at a point of the phase space.

7. Execute '`integ`' to calculate differential and total cross-sections with `BASES`.

8. Execute '`spring`' to get four-momenta of outgoing particles generated with `SPRING`.

At last, we can simulate event-generations in experiments of high-energy physics.

3 Specifications of a utility program COMPA

Here we consider a problem on the composite particles. The `GRACE` system is based on the elementary-particle processes. On the other hand, a proton (one of composite particles) consists of several quarks, anti-quarks and a gluon. Then there are several elementary-particle processes in the process $e + P \to e + P$. Furthermore, we have to enumerate initial states by hand when we specify a final state in the level of the elementary-particle process.

We have solved the problem above by constructing a utility program named `COMPA` (a COMposite Particle to Amplitudes). The specifications of `COMPA` is the following.

Input files

1. Definition of a composite particle (compo.def): Users can define a composite particle in this file as "*Name of a composite particle = Names of elementary particles*".
   ex) `Proton=␣u␣d` $\cdots$
   A space is used as a delimiter between names of elementary particles. It is possible to write a list of the names in several lines (see Appendix A.2).

2. Definition of a process (in.prc): Users can use the name of the composite particle defined in compo.def once in each line of the definition on the initial state and the final state (see Appendix A.3).
   ex)
   ```
   Initial={electron, Proton};
       Final={electron, Proton};
   ```

Generated files

1. Definition of processes (in.prc): Each input file for elementary-particle processes is generated in the corresponding sub-directory which is generated with a name of an elementary particle defined in compo.def (see Appendix B.1).
2. Log file (elements.log): The number of Feynman diagrams is recorded for each elementary process (see Appendix B.2).

Procedure in `COMPA`

1. Read the names of the composite particle and the elementary particles from compo.def.
2. Generate sub-directories with the names of the elementary particles.
3. Change the current directory to a sub-directory.
4. Read lines of in.prc one by one, and search the name of the composite particle.
5. Open a file in.prc in the current directory with assignment of the append mode, and write the lines of the original in.prc above to this file with replacing the name of the composite particle by that of an elementary particle.
6. Execute `grc` in the current directory.
7. Scan the generated file out.grf, open a file elements.log, and record the number of Feynman diagrams in it.
8. Execute `grcfort` to generate amplitudes if the number of diagrams is greater than zero.
9. Change the current directory to the parent directory.
10. Iterate the procedure from 2 to 9 above up to the end of the original in.prc.

4 Implementation

We have implemented the method described above as a computer program written in the `C` language on the `HP-UX9.07`. The source of `COMPA`, compa.c, is presented in Appendix C. The program is based on the `ANSI C` except for the access-control list in the system-call command `mkdir()` [26]. The executable file is easily obtained by the compiler on the `HP-UX` as follows:

```
cc -Ae -o compa compa.c
```

The option parameter '-Ae' represents the `ANSI C` with the `HP-UX` extensions. When users use another OS, they can obtain the executable file by rewriting `mkdir()` in compa.c and by compiling it with suitable option parameters.

We show an example of generated sub-directories and files in Fig. 2 for the process $e^- + P \to e^- + u + \bar{u} + X$. For this process, there exists only one elementary-partilcle

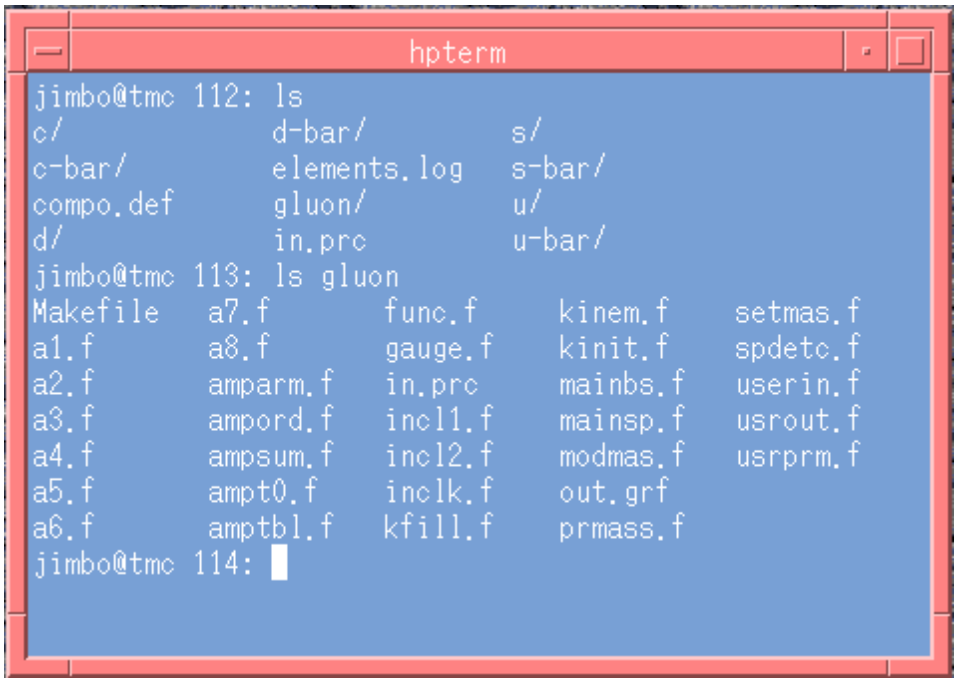

Figure 1: Fig. 2. For the process $e^- + P \to e^- + u + \bar{u} + X$

process $e^- + g \to e^- + u + \bar{u}$ (see Appendix B.2). The Feyman diagrams of this process are shown in Fig. 3.

5 Summary

The author has constructed a utility program `COMPA` to treat the processes including composite particles with the `GRACE` system which has been developed for the automatic computation of the matrix elements for the processes of the standard model and the minimal SUSY standard model (MSSM).

The program `COMPA` generates sub-directories and in.prc's of corresponding elementary-particle processes, and also execute `grc` and `grcfort` automatically. Only tasks left for us are constructing files of kinematics with the structure function and another utility program for binding elementary-particle processes.

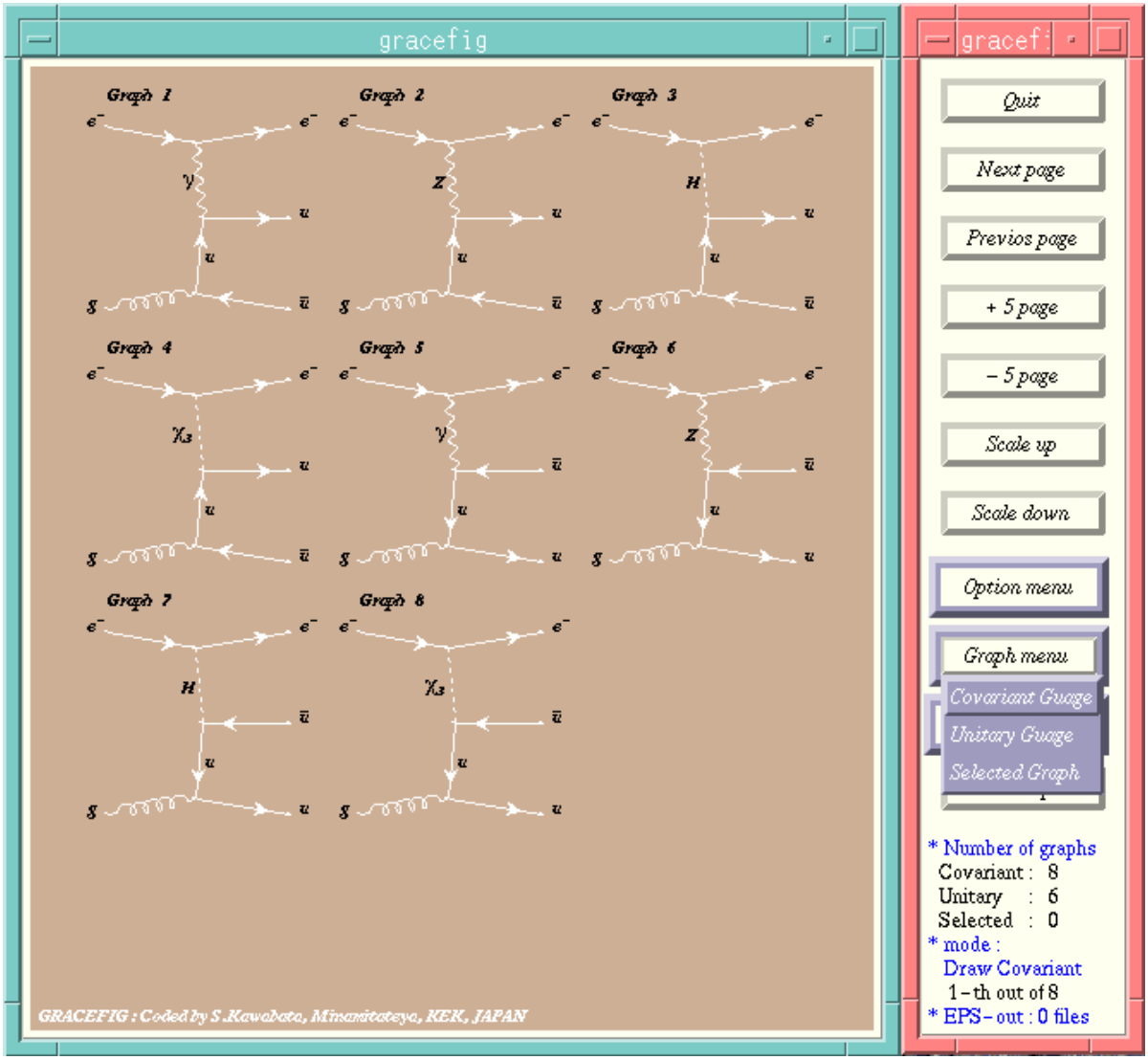

Figure 2: Fig. 3. Feyman diagrams of the process $e^- + g \to e^- + u + \bar{u}$

Acknowledgements

This work was supported in part by the Ministry of Education, Science and Culture (Mombu-sho), Japan under Grant-in-Aid for International Scientific Research Program No.07044097 and for Scientific Research (C) Program No.06640411, No.08640391 and No.08640400.

Appendix A Input files

A.1 'in.prc' as a usual case

The following example is a file '`in.prc`' specifying the tipical SUSY process $e^+ + e^- \to e^+ + \tilde{e}^+_{\rm R} + \tilde{\chi}^0_1$.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
model="/home/jimbo/grace/modelsusy/mssm.mdl";
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Process;
  ELWK=2;
  Initial={positron, electron};
  Final={positron, selectronR, neutralino1};
  Expand=Yes;
  OPI=Yes;
  Block=No;
  Selfe=Yes;
  Countert=No;
  Extself=No;
  Tadpole=No;
  Kinem="3001";
Pend;
```

A.2 'compo.def' for a composite particle

The following example is a file '`compo.def`' specifying the components of a comosite particle.

```
Proton= u d s c
u-bar d-bar s-bar c-bar
gluon
```

A.3 'in.prc' for a composite particle

The following example is a file '`in.prc`' specifying the process $e^- + P \to e^- + u + \bar{u} + X$.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
model="/home/jimbo/grace/modelelwk/all.mdl";
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Process;
  ELWK=2;
  QCD=1;
  Initial={electron, Proton};
  Final={electron, u, u-bar};
  Expand=Yes;
```

```
  OPI=Yes;
  Block=No;
  Selfe=Yes;
  Countert=No;
  Extself=No;
  Tadpole=No;
  Kinem="3001";
Pend;
```

Appendix B Generated files

B.1 'in.prc' generated by compa

The following example is a generated file '`in.prc`' specifying the process $e^- + g \to e^- + u + \bar{u}$.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
model="/home/jimbo/grace/modelelwk/all.mdl";
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Process;
  ELWK=2;
  QCD=1;
  Initial={electron, gluon};
  Final={electron, u, u-bar};
  Expand=Yes;
  OPI=Yes;
  Block=No;
  Selfe=Yes;
  Countert=No;
  Extself=No;
  Tadpole=No;
  Kinem="3001";
Pend;
```

B.2 'elements.log' generated by compa

The following example is a generated file '`elements.log`' recording the number of Feynman diagrams for the process $e^- + g \to e^- + u + \bar{u}$. The number '-1' indicates the lack of the corresponding elementary-particle process.

```
u -1
d -1
s -1
c -1
u-bar -1
```

```
    d-bar -1
    s-bar -1
    c-bar -1
    gluon 8
```

Appendix C The source file

The following is the source list of the program '`compa.c`'.

```
/*    compa.c             (a COMposite Particle to Amplitudes)

      Copyright (c)  Masato JIMBO 1997
      All Rights Reserved
*/

#include <string.h>       /* strcpy,strtok,size_t,memmove,strlen,strstr */
#include <sys/stat.h>     /* mkdir,chdir; <sys/stat.h> in HP-UX, */
                          /* <dir.h> in BC++, <direct.h> in QC */
#include <stdio.h>        /* FILE,NULL,fgets,fopen,fputs,printf */
#include <stdlib.h>       /* system */

#define sgrc "/home/jimbo/grace/bin/grc" /* depends on your environment */
#define sgrcfort "/home/jimbo/grace/bin/grcfort" /* depends on your env */
#define MAX_i 32          /* the maximum number of elements = MAX_i - 1 */
#define PERM S_IRWXU|S_IRGRP
                  /* the permission mode of the directories on the unix */
#define sPend "Pend"                 /* the number of graphs in out.grf */

char *repastr(char *, const char *, const char *);
int comp2el(void);

int main(void)
{
  int err;

  err=comp2el();

  switch(err)
  {
   case 1:
    printf("The file \"compo.def\" dose not exist in the current directory.\n");
    break;
   case 2:
    printf("The file \"in.prc\" dose not exist in the current directory.\n");
    break;
   case 3:
    printf("The number of elements is overflowed.\n");
    break;
   case 4:
    printf("The file \"elements.log\" cannot be created ");
    printf("in the current directory.\n");
    break;
   case 5:
    printf("Some sub-directory cannot be created in the current directory.\n");
    break;
   case 6:
    printf("The file \"in.prc\" cannot be created in some sub-directory.\n");
    break;
   default: printf("The process was finished successfully.\n");
  }
```

```
  return 0;
} /* the end of main */

int comp2el(void)    /* a COMposite Particle TO its ELements */
{
  FILE *f_in_def;
  FILE *f_in_pro;
  FILE *f_in_grf;
  FILE *f_out_pro;
  FILE *f_out_log;

  int err_flag=0,perm_err;
  int i,top_l,i_end,j;
  char s[256];          /* an array for one line */
  char s1[256];         /* an array for one line */
  char s2[256];         /* an array for one line */
  char sel[MAX_i][64]; /* arrays in which the tokens are saved */
  char *ptmp;           /* a temporary pointer for a token */
  char sgr[32];         /* an array in which the number of graphs is saved */

  f_in_def=fopen("compo.def","r");
  if(f_in_def!=NULL)
  {
    i=0;
    while(fgets(s,sizeof s,f_in_def)!=NULL)  /* read lines from compo.def */
    {
      top_l=0;
      repastr(s,"\n"," ");  /* replace a return code with a space */
      /* The delimiter of elements is a space. */
      if(i==0)
      {
        strcpy(sel[i],strtok(s,"="));
        top_l=-1;
      }       /* The name of the composite particle has to be placed */
              /* at the left of a sign of equality. */
      if(top_l==0)
      {
        i++;
        strcpy(sel[i],strtok(s," "));
        top_l=-1;
      }   /* for the first token in the line (on and after the second line) */
      if(top_l!=-1) break;
      while((ptmp=strtok(NULL," "))!=NULL)
      {
        i++;
        if(i>=MAX_i)
        {
          err_flag=3;
          break;
        }
        strcpy(sel[i],ptmp);
      }
    } /* the end of reading lines from compo.def */
    i_end=i;
    fclose(f_in_def);
    for(j=1;j<=i_end;j++)
    {
      f_in_pro=fopen("in.prc","r");
      if(f_in_pro!=NULL)
      {
        f_out_log=fopen("elements.log","a");
        if(f_out_log==NULL)
        {
```

```
          err_flag=4;
          break;
        }
        perm_err=mkdir(sel[j],PERM);
        if(perm_err==-1)
        {
          err_flag=5;
          break;
        }
        chdir(sel[j]);
        f_out_pro=fopen("in.prc","w");
        if(f_out_pro!=NULL)
        {
          while(fgets(s1,sizeof s1,f_in_pro)!=NULL) /* read lines from in.prc */
          {
            repastr(s1,sel[0],sel[j]); /* replace the first token with another */
            fputs(s1,f_out_pro);       /* write a line into the new in.prc */
          }
          fclose(f_out_pro);
          system(sgrc);
          strcpy(s2,sel[j]);
          strcpy(sgr,"0");
          f_in_grf=fopen("out.grf","r");
          if(f_in_grf!=NULL)
          {
            while(fgets(s,sizeof s,f_in_grf)!=NULL) /* read lines from out.grf */
            {
              if(strstr(s,sPend)!=NULL)
              {
                strtok(s,"=");
                strcpy(sgr,strtok(NULL,";"));
              }
            }
            fclose(f_in_grf);
            if(strcmp(sgr,"0")>0) system(sgrcfort);
            strcat(s2," ");
            strcat(s2,sgr);
            strcat(s2,"\n");
          }
          else strcat(s2," -1\n");
          fputs(s2,f_out_log);
        }     /* the end of if(f_out_pro!=NULL) */
        else err_flag=6;
        chdir("..");
        fclose(f_out_log);
        fclose(f_in_pro);
      }        /* the end of if(f_in_pro!=NULL) */
      else err_flag=2;
    }               /* the end of the for loop */
  }            /* the end of if(f_in_def!=NULL) */
  else err_flag=1;

return err_flag;
} /*  the end of comp2el */

char *repastr( /* REPlace A STRing with another */
               char      *st,   /* Total String */
               const char *so,    /* Old String */
               const char *sn)    /* New String */
{
  size_t Lo;   /* Length of so */
  size_t Ln;   /* Length of sn */
  char *const p=strstr(st,so);    /* the address of the "so" found in the "st" */
```

```
  if(p==NULL) return NULL; /*if the "so" is not found in the "st", return NULL.*/

  Lo=strlen(so);
  Ln=strlen(sn);

  if(Lo>=Ln)
  {
    memmove(p,sn,Ln);
    if(Lo>Ln) memmove(p+Ln,p+Lo,st+strlen(st)-(p+Lo-1));
  }
  else
  {
    memmove(p+Ln,p+Lo,st+strlen(st)-(p+Lo-1));
    memmove(p,sn,Ln);
  }

  return p+Ln;  /* the address of the point at which */
                /* the last replacement occured */
} /* the end of repastr */
```

References

[1] C. Dionisi, in *Proceedings of XVII International Meeting on Fundamental Physics, PHYSICS AT LEP*, Lekeitio, April 23-29, 1989, edited by M.A.-Benítez and M. Cerrada, (World Scientific, Singapore, 1990), p.71.
ALEPH Collaboration, D. Decamp *et al.*, *Phys. Lett.* **244B** (1990), 541.
DELPHI Collaboration, P. Abreu *et al.*, *Phys. Lett.* **247B** (1990), 157.
Proceedings of the Joint International Lepton-Photon Symposium & Europhysics Conference on High Energy Physics, Geneva, Switzerland, July 25-August 1, 1991, edited by S. Hegarty, K. Potter and E. Quercigh, (World Scientific, Singapore, 1992).

[2] *Proceedings of the workshop on Physics at Future Accelerators*, La Thuile and CERN, January 1987, edited by J.H. Mulvey, CERN Report *CERN* **87-07** (1987).
C. Ahn *et al.*, *SLAC-Report*-**329** (1988).
R. Barbieri *et al.*, *Z PHYSICS AT LEP 1*, CERN Report *CERN* **89-08** Vol.**2** (1989), p.121.
Proceedings of the Third Workshop on Japan Linear Collider (JLC), KEK, February 18-20, 1992, edited by A. Miyamoto, *KEK Proceedings* **92-13** (1992).

[3] H. Bear *et al.*, preprint *FSU-HEP*-**950401** (*LBL*-**37016**, *UH*-**511-822-95** and *hep-ph/* **9503479**) (1995), and References therein.

[4] H. Tanaka, T. Kaneko and Y. Shimizu, *Comput. Phys. Commun.* **64** (1991), 149.

[5] I. Watanabe, H. Murayama and K. Hagiwara, in *Proceedings of the Third Workshop on Japan Linear Collider (JLC)*, KEK, February 18-20, 1992, edited by A. Miyamoto, *KEK Proceedings* **92-13** (1992), p.265.

[6] H. Tanaka, *Comput. Phys. Commun.* **58** (1990), 153.

[7] T. Kaneko, in *New Computing Techniques in Physics Research*, edited by D. Perret-Gallix and W. Wojcik, (Édition du CNRS, Paris, 1990), p.555.
T. Kaneko and H. Tanaka, in *Proceedings of the Second Workshop on Japan Linear*

Collider (JLC), KEK, November 6-8, 1990, edited by S. Kawabata, *KEK Proceedings* **91-10** (1991), p.250.
T. Kaneko, in *New Computing Techniques in Physics Research II*, edited by D. Perret-Gallix, (World Scientific, Singapore, 1992), p.659.
T. Ishikawa, T. Kaneko, K. Kato, S. Kawabata, Y. Shimizu and H. Tanaka (Minami-Tateya group), *GRACE manual Version 1.0*, *KEK Report* **92-19** (1993), and References therein.

[8] E. Boos, M. Dubinin, V. Edneral, V. Ilyin, A. Kryukov, A. Pukov, V. Savrin, S. Shichanin and A. Taranov, in *New Computing Techniques in Physics Research*, edited by D. Perret-Gallix and W. Wojcik, (Édition du CNRS, Paris, 1990), p.573.
E. Boos, M. Dubinin, V. Edneral, V. Ilyin, A. Kryukov, A. Pukov, S. Shichanin, in *New Computing Techniques in Physics Research II*, edited by D. Perret-Gallix, (World Scientific, Singapore, 1992), p.665.
A. Pukov, in *New Computing Techniques in Physics Research III*, edited by K.-H. Becks and D. Perret-Gallix, (World Scientific, Singapore, 1994), p.473.

[9] J. Küblbeck, M. Böhm and A. Denner, *Comput. Phys. Commun.* **60** (1990), 165.
R. Mertig, M. Böhm and A. Denner, *Comput. Phys. Commun.* **64** (1991), 345.
R. Mertig, in *New Computing Techniques in Physics Research III*, edited by K.-H. Becks and D. Perret-Gallix, (World Scientific, Singapore, 1994), p.467.
H. Eck and J. Küblbeck, *ibid.*, p.565.

[10] T. Stelzer and W.F. Long, *Comput. Phys. Commun.* **81** (1994), 357.

[11] A. Denner, H. Eck, O. Hahn and J. Küblbeck, *Phys. Lett.* **B 291** (1992), 278.
A. Denner, H. Eck, O. Hahn and J. Küblbeck, *Nucl. Phys.* **B 387** (1992), 467.

[12] S. Kawabata, *Comput. Phys. Commun.* **41** (1986), 127.
S. Kawabata, *Comput. Phys. Commun.* **88** (1995), 309.

[13] M. Jimbo and H. Tanaka, *Talk presented at JPS meeting*, Fukuoka, March 28-31, 1994.

[14] M. Jimbo, H. Tanaka, T. Kaneko, T. Kon and Minami-Tateya collaboration, in *Physics of e^+e^-, $e^-\gamma$ and $\gamma\gamma$ collisions at linear accelerators — Proceedings of the INS Workshop*, INS, December 20-22, 1994, edited by Z. Hioki, *et al.*, *INS-J*-**181** (1995), p.222.

[15] M. Jimbo, T. Kon and Minami-Tateya collaboration, in *Proceedings of the YITP Workshop on Particle Physics and its Future Perspective*, YITP, January 17-20, 1995, edited by K. Suehiro, *Soryushiron Kenkyu* **92** (1995), p.31.

[16] M. Jimbo and Minami-Tateya collaboration, in *Proceedings of the Fifth Workshop on Japan Linear Collider (JLC)*, Kawatabi, Miyagi, February 16-17, 1995, edited by Y. Kurihara, *KEK Proceedings* **95-11** (1995), p.98.

[17] T. Kon, T. Kaneko, H. Tanaka, M. Jimbo and Minami-Tateya collaboration, in *ELECTROWEAK INTERACTIONS AND UNIFIED THEORIES — Proceedings of XXXth Rencontres de Moriond*, Les-Arcs, Savoie, France, March 11-18, 1995, edited by J. Tran Thanh Van, (Éditions Fronties, Gif-sur-Yvette Cedex, 1996), p.287.

[18] M. Jimbo, T. Kon, H. Tanaka, T. Kaneko and Minami-Tateya collaboration, in *New Computing Techniques in Physics Research IV — Proceedings of the Fourth International Workshop on Software Engineering, Artificial Intelligence and Expert Systems for High Energy and Nuclear Physics (AIHENP95)*, Pisa, Italy, April 3-8, 1995, edited by B. Denby and D. Perret-Gallix, (World Scientific, Singapore, 1995), p.149.

[19] T. Kon, T. Kaneko, M. Jimbo, H. Tanaka and Minami-Tateya collaboration, in *Proceedings of the Workshop on Physics and Experiments with Linear Colliders*, Morioka-Appi, Iwate, Japan, September 8-12, 1995, edited by A. Miyamoto *et al.*, (World Scientific, Singapore, 1996), p.579.

[20] H.P. Nilles, *Phys. Rep.* **110** (1984), 1.
H.E. Haber and G.L. Kane, *Phys. Rep.* **117** (1985), 75.
M. Chen, C. Dionisi, M. Martinez and X. Tata, *Phys. Rep.* **159** (1988), 201.
R. Barbieri, *Riv. Nuovo Cimento* **11** (1988).

[21] H. Tanaka, M. Kuroda, T. Kaneko, M. Jimbo, T. Kon and Minami-Tateya collaboration, *Nucl. Inst. Meth. Phys. Res.* **A389** (1997), 295.

[22] T. Ishikawa, T. Kaneko, K. Kato, S. Kawabata, Y. Shimizu and H. Tanaka, *Grace Manual Version 1.0, KEK Report* **92-19** (1993).
T. Ishikawa, S. Kawabata, Y. Kurihara and , *Brief Manual of Grace System Version 2.0 β* (1995), unpublished.

[23] T. Kaneko, in *New Computing Techniques in Physics Research IV — Proceedings of the Fourth International Workshop on Software Engineering, Artificial Intelligence and Expert Systems for High Energy and Nuclear Physics (AIHENP95)*, Pisa, Italy, April 3-8, 1995, edited by B. Denby and D. Perret-Gallix, (World Scientific, Singapore, 1995), p.313.

[24] T. Kaneko, *Comput. Phys. Commun.* **92** (1995), 127.

[25] T. Ishikawa, S. Kawabata and Y. Kurihara, in *Proceedings of the Fifth Workshop on Japan Linear Collider (JLC)*, Kawatabi, Miyagi, February 16-17, 1995, edited by Y. Kurihara, *KEK Proceedings* **95-11** (1995), p.92.

[26] Hewlett-Packard Company, *HP-UX 9.0 Reference, HP Part No.* **B2355-90033** (1992).