De JurassicFortran à Fortr'&Furious

Vincent Lafage

IJCLab, CNRS/IN2P3 & Université Paris-Saclay, Orsay, France

10 juillet 2023

- 1954 : *Automatic Coding System for the IBM 704* dated October 15, 1956, first presented in February 1957 and first delivered in April 1957
- FORmula TRANslator : langage de haut niveau gérant la première abstraction, l'abstraction des expressions :
  Le compilateur transforme les expressions en arbre d'évaluation
- pas le plus ancien compilateur, mais presque, et en tous cas le plus ancien à subsister
- le plus ancien compilateur optimiseur
- format fixe (cartes perforées)
- orienté calcul et pas système (relativement haut niveau = abstraction vis-à-vis du modèle mémoire)
- peu de types économie de moyen déclarations implicites

Historique

- 1966, *a.k.a.* `FORTRAN IV` première standardisation (ANSI X3.9-1966)
  (paléo FORTRAN)
- 1977 (ANSI X3.9-1978 FORTRAN, ISO 1539 :1980) influence de la programmation structurée :
  (archéo FORTRAN)
  paradigme **procédural** langage de haut niveau gérant la deuxième abstraction, l'abstraction du flux de contrôle
- 1989 `f2c` premier traducteur open source de `FORTRAN` vers `C`
- 1988 ? Rendez-vous manqué avec l'histoire
- 1991 `FORTRAN` ⇒ `Fortran`
  `Fortran 90` : langage de haut niveau gérant la troisième abstraction, l'abstraction des données
  paradigme **modulaire** + gestion des types dérivés `struct/record`
- 1995, Feb 17 : `g77` premier compilateur open source de `FORTRAN`
- *Free Format*
- 1997 `Fortran 95` : allocation dynamique
- early 2000—2006 : `g95` premier compilateur open source de `Fortran`
- 2003, Jan—2005, Apr 20 : `gfortran` fork de `g95`

Missing In Fortran

- héritage multiple ⇒ Facade Pattern
- first-class functions ⇒ `real, external, pointer :: f_ptr`.
  Alternative : `procedure (f), pointer :: f_ptr`
  `https://gcc.gnu.org/onlinedocs/gfortran/Working-with-C-Pointers.html`
- gestion d'exception (au-delà des exceptions IEEE 754)
- `assert`
- généricité (polymorphisme paramétrique) ⇒ `Fortran202Y`
  `https://github.com/j3-fortran/generics/tree/main`
- `const` ? (≠ `parameter` ∼ `consteval`) ⇒ …, `intent (in) ::` ou …, `protected ::`
- `C` binding to variadic functions
- coroutines
- …

Fortran programming is not FORTRAN programming

Aspect sociolinguistique

- Style (supposé) de FORTRAN :
  - format fixe
  - pas d'indentation
  - ALL CAPS !
  - identifiant trop courts (6 caractères)
  - typage implicite
  - répétitions des déclarations et des COMMON
  - fonctions trop longues
  - GOTO à outrance, GOTO arithmétique, code spaggheti
- ⇒ AdaTran (DOI:10.1145/126551.126611) : de l'Ada écrit (supposément) comme du FORTRAN

https://jackkrupansky.medium.com/what-is-adatran-or-adatran-or-adatran-6bab09c44b93

1. The use of the Ada programming language as if it were the FORTRAN programming language.
2. Coding in the syntax of Ada with the style and semantics of FORTRAN.
3. Ada code written by a FORTRAN programmer.
4. A subset of Ada taught to FORTRAN programmers.
5. Ada code generated by an automated translation of FORTRAN code.
6. Ada code generated by automated translation of any non-Ada programming language.
7. The use of any advanced technology as if it were an older technology with which one is more familiar and experienced.

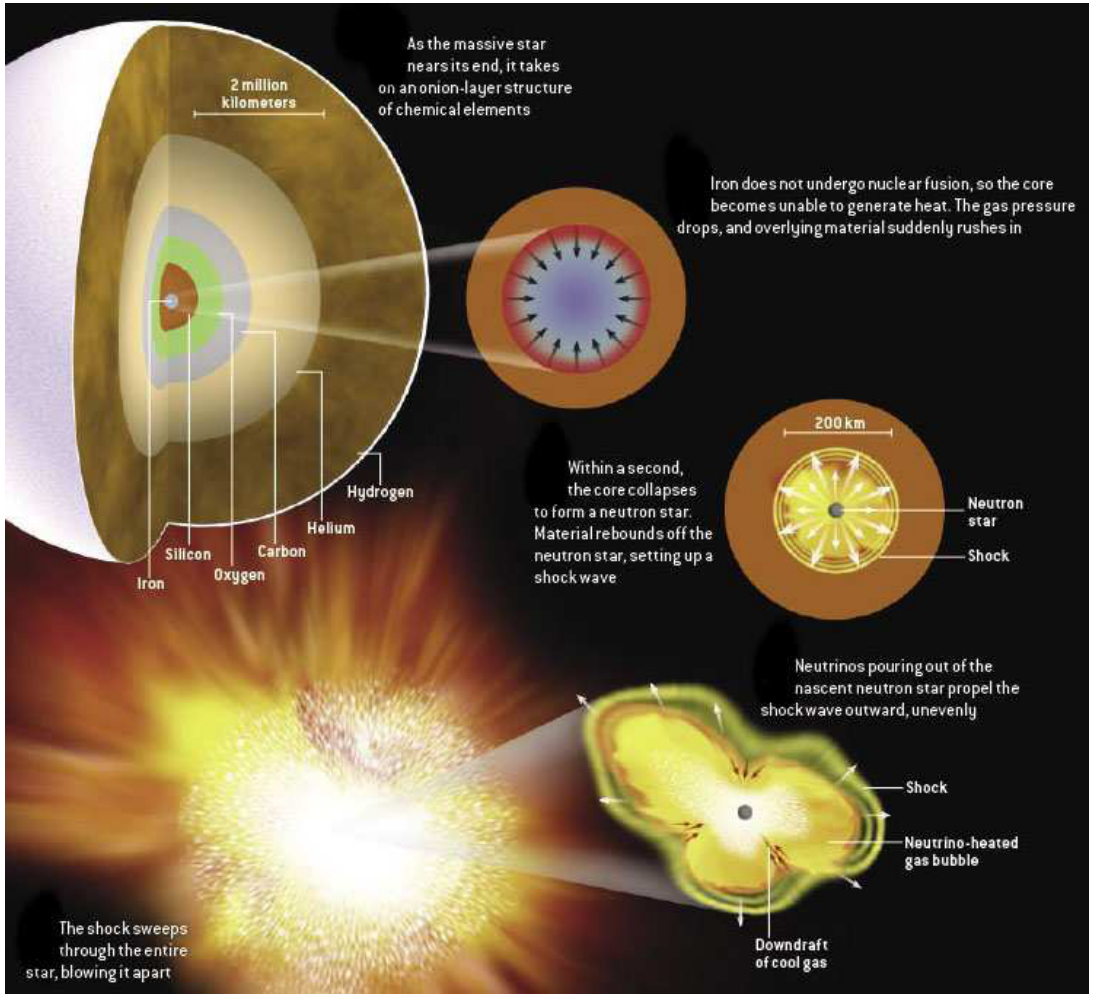
`Fortran` in the era of Software Engineering

- modern editor
- despagghetifier of legacy code
- version control
- build system
- a bunch of static analysis tools

Capture d'électrons & supernovae

Groupe théorie IPNO
Simulation :
1000 types de noyaux
50 valeurs de T
$\Rightarrow$ 1000 ans de CPU

* *profiling*
* audit bibliothèques
* gestion mémoire
* vectorisation



$\Rightarrow$ **facteur $>$ 55**

$\Rightarrow$ **20 ans de CPU après accélération**

Structure du programme

```
(... boucles sur J, Π ...)                ! 6
(... calcul de structure nucléaire selon un modèle RPA : ...)
(... occupations des niveaux d'énergies, fonctions d'ondes wf / dwf ...)
do idxenergy0 = 0, nbenergystep - 1       ! 150 énergies electron
   do energyc = 1, nconf                  ! 153 / 405 / 540 états d'énergie du noyau
      do k_simps = 0, n_simps             ! 31 angles electron-neutrino
         do pairc = 1, nconf              ! 153 / 405 / 540 états d'énergie du noyau T < 1 MeV
                                          ! 182 / 476 / 637 états d'énergie du noyau T = 2 MeV
                                          ! 232 / 609 / 821 états d'énergie du noyau T = 3 MeV
                                          ! 386 / 1041 / 1448 états d'énergie du noyau T = 4 MeV
            do i = 1, maxr                ! 168 mailles radiales de fonction d'onde…
               à fond de boucles, fonctions de bessel ?sphériques> 92% du temps
```

$$j_\ell(qr)$$

nid de 5 boucles indépendantes

$\Longrightarrow$ paradis des parallélistes

entre 121 millions et 11 milliards d'itérations seulement pour les 4 boucles internes

OpenACC

```
subroutine radproint_array (j0)
  implicit none
  integer, intent (in) :: j0
  integer :: k_simps, pairc, idxenergy, energyc, maj ! dummy indices
  integer :: Jmin, Jmax
  Jmin = max (j0-1, 0)
  Jmax = j0+1

  allocate (Rad_proint1_array  (1:nconf, 0:n_simps, 1:nconf, 0:nbenergystep-1, Jmin:Jmax))

  !$acc data present_or_create (mask_kinematics, Rad_proint2B_array, Rad_proint2A_array, Rad_pro
  !$acc                        wf12_array, Bess_f_array, wf12inv_array, wf1dwf2_array)

  !$acc parallel loop collapse (3)
  build10: do maj = Jmin, Jmax
    build11: do idxEnergy = 0, nbenergystep-1
      build12: do energyc = 1, nconf
        if (mask_kinematics (energyc, idxEnergy)) then
  !$acc loop vector independent collapse (2)
          build13: do k_simps = 0, n_simps
            build14: do pairc = 1, nconf
              Rad_proint1_array (pairc, k_simps, energyc, idxEnergy, maj) = &
              sum (wf1dwf2_array (1:maxr, pairc) * Bess_f_array (1:maxr, k_simps, energyc, idxEn
            end do build14
          end do build13
        end if
      end do build12
    end do build11
  end do build10
  !$acc end parallel loop

  !$acc parallel
```

Speedup (news)

	2013 Intel	r88 Intel	r153 F90 Intel	r242 new Intel	Intel ×	GNU	GNU ×	PGI	PGI ×	PGI OpenACC	PGI OpenACC
ftchrpa	39,31 42,77	39,31	39,31	39,31	39,14	83,19	83,15	69,35	69,27	69,41	
eccalc	502,05	10,21	19,17	4,66	5,54	10,90	6,02	5,28	5,95	4,66	5,04
total	541	49,52	58,47	44,01	44,73	94,14	89,21	83,22	83,88	82,67	53,83
×eccalc		49,2		107,8	90,7		83,4		84,4	107,8	
×eccalcr.				2,2	1,8		1,7		1,7	2,2	

Conclusion

- beyond OpenACC
- Auto GPUfication
- Nouveaux CUDA
- Nouveau GPU