Comment compiler. Compilateurs Intel.

Comment compiler. Compilateurs Intel.

- Opération de compilation
- Serveurs de compilation
- Compilateurs installés
- Compilateurs Intel
- Débogage

Opération de compilation

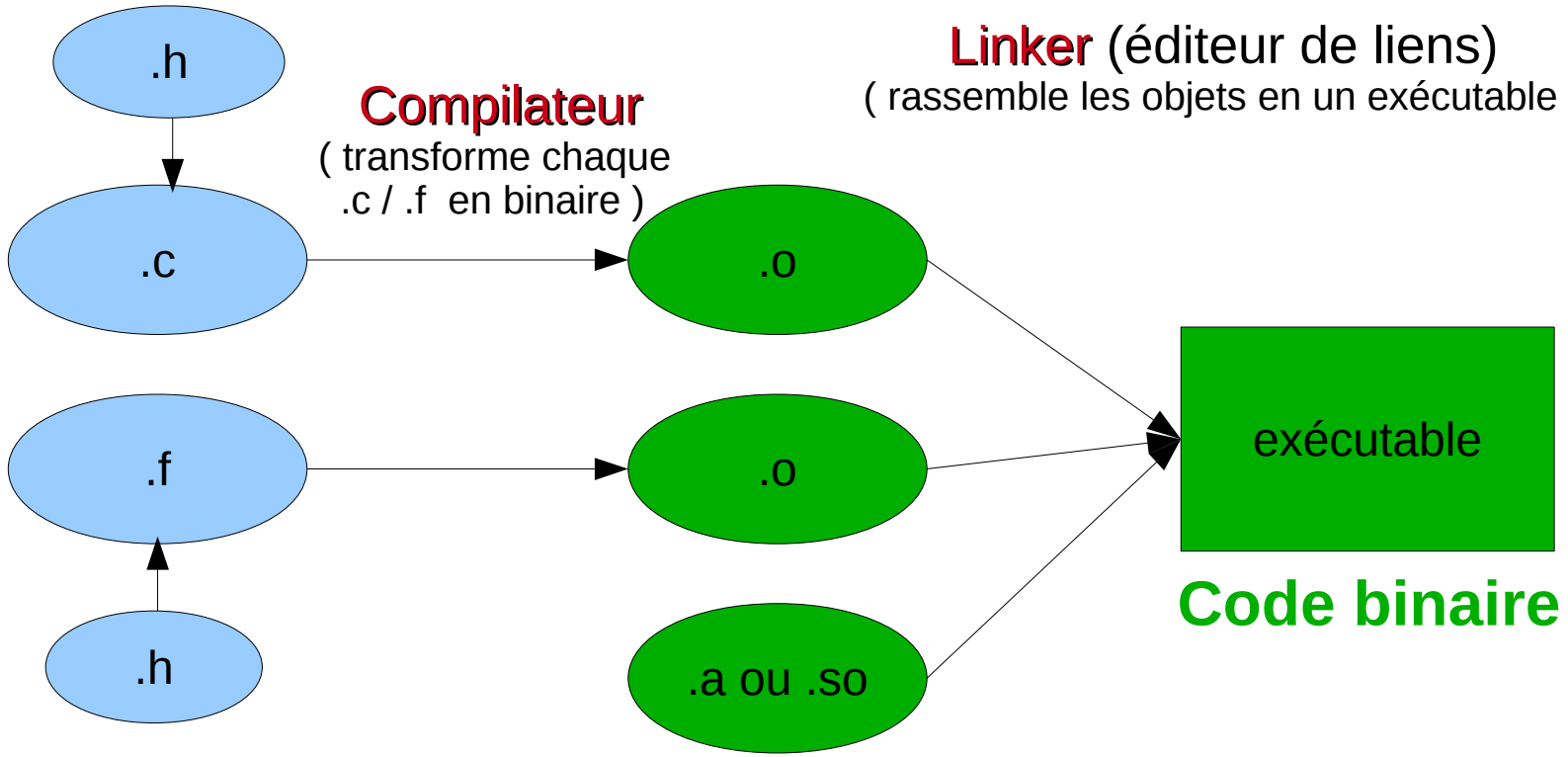
Compilation : processus de transformation d'un **code source** en un **code binaire**

Compilation : processus de transformation d'un **code source** en un **code binaire**

Phases principales

Préprocesseur

(inclusion des .h dans les .c et/ou .f)



Serveurs de compilation

- Pour compiler un code qui sera exécuté sur un groupe de serveurs avec **proc intel**
  - ➔ Il vaut mieux le compiler sur un serveur de compilation avec **proc intel**

- Pour compiler un code qui sera exécuté sur un groupe de serveurs avec **proc AMD**
  - ➔ Il vaut mieux le compiler sur un serveur de compilation avec **proc AMD**

- Pour compiler un code qui sera exécuté sur un groupe de serveurs avec **proc intel**
  - ➔ Il vaut mieux le compiler sur un serveur de compilation avec **proc intel**

Groupe de serveurs	Processeur Intel Xeon	Serveur de compilation
c6100lin24	Westmere (6 coeurs 2,66 Ghz)	c6100comp-pub
r410lin24A r410lin24B r410lin24C	Nehalem (4 coeurs 2,66 Ghz)	r410comp1-pub r410comp2-pub
x41zlin16 x41zlin32	Harpertown (4 coeurs 2,66 Ghz)	x41zcomp-pub

- Pour compiler un code qui sera exécuté sur un groupe de serveurs avec **proc AMD**
  - ➔ Il vaut mieux le compiler sur un serveur de compilation avec **proc AMD**

- Pour compiler un code qui sera exécuté sur un groupe de serveurs avec **proc intel**
  - ➔ Il vaut mieux le compiler sur un serveur de compilation avec **proc intel**

Groupe de serveurs	Processeur Intel Xeon	Serveur de compilation
c6100lin24	Westmere (6 coeurs 2,66 Ghz)	c6100comp-pub
r410lin24A r410lin24B r410lin24C	Nehalem (4 coeurs 2,66 Ghz)	r410comp1-pub r410comp2-pub
x41zlin16 x41zlin32	Harpertown (4 coeurs 2,66 Ghz)	x41zcomp-pub

- Pour compiler un code qui sera exécuté sur un groupe de serveurs avec **proc AMD**
  - ➔ Il vaut mieux le compiler sur un serveur de compilation avec **proc AMD**

Groupe de serveurs	Processeur AMD Opteron	Serveur de compilation
v40zlin4ib v22zlin4ib	AMD Opteron	v22zcomp1-pub v22zcomp2-pub
dl165lin24	Shangaï (4 coeurs 2,66 Ghz)	dl165comp-pub
dl175lin32	Istambul (6 coeurs 2,66 Ghz)	dl175comp-pub
r815lin128	Magny-Cours (8 coeurs 2,3 Ghz)	r815comp-pub

Connexion à un serveur de compilation :

- Connexion sur allo-psmn.ens-lyon.fr

```
ssh -X votrelogin@allo-psmn.ens-lyon.fr
```
(réseau ENS)

Connexion à un serveur de compilation :

- Connexion sur allo-psmn.ens-lyon.fr

```
ssh -X votrelogin@allo-psmn.ens-lyon.fr
```
(réseau ENS)

- Connexion sur l'un de serveurs de compilation

```
ssh -X c6100comp-pub
ssh -X r410comp1-pub
ssh -X r410-comp2-pub
ssh -X x41zcomp-pub
ssh -X dl165comp-pub
ssh -X dl175comp-pub
ssh -X r815comp-pub
```

Connexion à un serveur de compilation :

• Connexion sur allo-psmn.ens-lyon.fr

```
ssh -X votrelogin@allo-psmn.ens-lyon.fr
```
(réseau ENS)

• Connexion sur l'un de serveurs de compilation

```
ssh -X c6100comp-pub
ssh -X r410comp1-pub
ssh -X r410-comp2-pub
ssh -X x41zcomp-pub
ssh -X dl165comp-pub
ssh -X dl175comp-pub
ssh -X r815comp-pub
```

Attention !

• La compilation de programmes se fait sur les serveurs de compilation.

Connexion à un serveur de compilation :

• Connexion sur allo-psmn.ens-lyon.fr

```
ssh -X votrelogin@allo-psmn.ens-lyon.fr
```
(réseau ENS)

• Connexion sur l'un de serveurs de compilation

```
ssh -X c6100comp-pub
ssh -X r410comp1-pub
ssh -X r410-comp2-pub
ssh -X x41zcomp-pub
ssh -X dl165comp-pub
ssh -X dl175comp-pub
ssh -X r815comp-pub
```

Attention !

• La compilation de programmes se fait sur les serveurs de compilation.

•L'exécution de programmes se fait uniquement via une soumission au système de batch, sauf pour un test rapide

Compilateurs installés

Nombre de licences par type de compilateur

Type de compilateur	Nombre de jetons
INTEL	illimité pour l'instant → 2
GNU	illimité (gratuit)
PGI (Portland Group)	2 , exécution limitée à 16 CPUs
ps (Pathscale)	2 , exécution illimitée

Nombre de licences par type de compilateur

Type de compilateur	Nombre de jetons
INTEL	illimité pour l'instant → 2
GNU	illimité (gratuit)
PGI (Portland Group)	2 , exécution limitée à 16 CPUs
ps (Pathscale)	2 , exécution illimitée

Remarques :

- Le nombre limité de jetons ne concerne que l'opération de compilation et pas l'exécution des programmes.
- PGI : exécution limitée à 16 coeurs
- Ces compilateurs sont installés dans ***/softs***

Les compilateurs séquentiels

	Nom	Chemin d'accès	Dernières versions installées	Fichier d'environnement dans **/usr/local/modeles**
INTEL	ifort icc/icpc	/softs/intel/version/bin/intel64	10.1.015 11.1.069	intel-version
GNU	*g77* gfortran gcc/g++	/usr/bin	*3.4.6* 4.2.1	openmpi-gnu
PGI	pgcc pgCC pgf77 pgf90 pgf95	/softs/pgi/linux86-64/version/bin	9.0-2 10.3 10.4	pgi-version
Ps	pathcc pathCC pathf90 pathf95	/softs/pathscale/bin	3.2	ps-3.2

Les compilateurs séquentiels

Exemple :

- Initialisation de l'environnement pour les compilateurs INTEL version 11.1.069

On exécute :

```
c6100lin37%  source /usr/local/modeles/intel-11.1.069
```

Alors les chemins d'accès pour les commandes ifort, icc, icpc seront activés et les chemins pour trouver les bibliothèques sont définis :

Les compilateurs séquentiels

Exemple :

- Initialisation de l'environnement pour les compilateurs INTEL version 11.1.069

On exécute :

```
c6100lin37%  source /usr/local/modeles/intel-11.1.069
```

Alors les chemins d'accès pour les commandes ifort, icc, icpc seront activés et les chemins pour trouver les bibliothèques sont définis :

Ex1 :
```
c6100lin37%  ifort -v
             Version 11.1
```

Ex2 :
```
c6100lin37%  man icc
ICC(1)  Intel(R) C++ Compiler Options  ICC(1)
   icc - invokes the Intel(R) C++ Compiler
.....................................
```

Les compilateurs séquentiels

Exemple :

- Initialisation de l'environnement pour les compilateurs INTEL version 11.1.069

On exécute :

```
c6100lin37%  source /usr/local/modeles/intel-11.1.069
```

Alors les chemins d'accès pour les commandes ifort, icc, icpc seront activés et les chemins pour trouver les bibliothèques sont définis :

Ex1 :
```
c6100lin37%  ifort -v
             Version 11.1
```

Ex2 :
```
c6100lin37%  man icc
ICC(1)  Intel(R) C++ Compiler Options  ICC(1)
   icc - invokes the Intel(R) C++ Compiler
.....................................
```

Ex3 :
```
c6100lin37% echo $LD_LIBRARY_PATH
/softs/intel/v11.1.069/mkl/lib/em64t:
/softs/intel/v11.1.069/lib/intel64:
/usr/local/sge/lib/lx24-amd64
```

Les compilateurs parallèles

- Ce sont des compilateurs générés en utilisant OpenMPI avec les compilateurs séquentiels précédents.

Les compilateurs parallèles

- Ce sont des compilateurs générés en utilisant OpenMPI avec les compilateurs séquentiels précédents.

	Nom	Chemin d'accès	Dernières versions OpenMPI	Scripts de configuration dans **/usr/local/modeles**
INTEL	mpif77	/softs/openmpi-version-intel-version	1.2.8 1.3.3 1.4.1	openmpi-intel
GNU	mpif90	/softs/openmpi-version-gnu-version		openmpi-gnu
PGI	mpicc	/softs/openmpi-version-pgi-version		openmpi-pgi
Ps	mpiCC	/softs/openmpi-version-ps-version		openmpi-ps

Les compilateurs parallèles

Exemple :

- Initialisation de l'environnement pour les compilateurs parallèles générés par Openmpi 1.4.1 avec le compilateur séquentiel INTEL version 11.1.069

On exécute :

```
c6100lin37% source /usr/local/modeles/openmpi-1.4.1-intel-11.1.069
```

En plus de commandes ifort, icc, icpc (séquentiels),
les chemins d'accès pour mpif77, mpif90, mpifcc, mpifCC seront activés
(et les chemins pour trouver les bibliothèques sont définis) :

Les compilateurs parallèles

Exemple :

- Initialisation de l'environnement pour les compilateurs parallèles générés par Openmpi 1.4.1 avec le compilateur séquentiel INTEL version 11.1.069

On exécute :

```
c6100lin37% source /usr/local/modeles/openmpi-1.4.1-intel-11.1.069
```

En plus de commandes ifort, icc, icpc (séquentiels),
les chemins d'accès pour mpif77, mpif90, mpifcc, mpifCC seront activés
(et les chemins pour trouver les bibliothèques sont définis) :

Ex :
```
c6100lin37%  which mpif90
/softs/openmpi-1.4.1-intel-11.1.069/bin/mpif90
```

Compilateurs Intel

Utilisation des compilateurs Intel

1. Connexion sur un serveur de compilation

2. Initialisation des variables d'environnement

3. Compilation et/ou édition de liens

Remarque

- *Avec un «compilateur» on peut :*
  - compiler et éditer les liens en une seule étape
  ou
  - compiler la source et ensuite éditer les liens (en deux étapes)

- *Pour compiler sans éditer les liens, l'option -c doit être utilisée, autrement les deux opérations sont faites*

Compilation et/ou édition de liens des programmes séquentiels

- **Compilateur Fortran**

  ifort [options] input_file(s)

- **Compilateur C / C++**

  icc [options] input_file(s)

où input_files peuvent être :

- des fichier source : – .f .for .ftn .f90 .F .FOR .F90 (pour Fortran et ifort)
  – .C .c .cc .cpp .cxx .c++ (pour C / C++ et icc)

- des fichiers objet : .o

- des bibliothèques statiques : .a

Options usuelles

Option	Description	Défaut
-c	Effectue uniquement la compilation. Un fichier objet (.o) est produit	OFF
-o nom_de_fichier	Spécifie le nom de l'exécutable produit. Si elle n'est pas utilisée, l'exécutable sera a.out	OFF
-O0,-O1,-O2,-O3	Spécifie le niveau d'optimisation (-O0 désactive tout optimisation)	-O2
-Bdynamic	Permet la liaison dynamique des bibliothèques au moment de l'exécution	OFF
-Bstatic	Permet la liaison statique des bibliothèques d'un utilisateur	OFF
-static	Empêche l'édition de lien avec les bibliothèques partagées	ON
-g	Produit des informations de débogage dans le fichier objet (exploitées par le débugger)	OFF

Options usuelles

Option	Description	Défaut
-I*dir*	Ajoute le répertoire dir à la liste des répertoires contenant les fichiers modules et includes	OFF
-L*dir*	Spécifie à l'éditeur de lien où chercher les bibliothèques avant de chercher dans les répertoires standard	OFF
-l*nom*	Spécifie à l'éditeur de lien de chercher la bibliothèque *libnom* lors de l'édition de liens	OFF
-i{2 | 4 | 8}	Spécifie la valeur KIND par défaut pour les variables entières et logiques	-i4
-m32 **-m64**	Indique au compilateur de générer du code pour l'architecture 32 bits ou 64 bits (par défaut, la compilation se fait selon l'architecture du serveur de compilation)	OFF
-mp1	Permet d'améliorer la précision et la cohérence en virgule flottante	OFF
-openmp	Permet la génération de code parallélisé multithread basé sur les directives OpenMP	OFF

Options usuelles

Option	Description	Défaut
-V	Affiche des information sur la version du compilateur	OFF
-p	Compile et édite les liens en fournissant des informations pour le profilage avec gprof	OFF
-fast	Maximise la vitesse à travers l'ensemble du programme (attention danger : les options qui sont implicitement invoquées peuvent changer d'une version à l'autre de compilateur)	OFF
-shared-intel	Lie dynamiquement les bibliothèques fournies par Intel	OFF
-w	Spécifie au compilateur de n'afficher aucun message d'avertissement	OFF
-shared	Spécifie au compilateur de produire un objet dynamique partagé (so) au lieu d'un exécutable	OFF

Compilation et/ou édition de liens des programmes séquentiels
Exemples

1) Pour compiler un programme Fortran 90 :

```
% ifort -c myprog.f90
```

2) Pour compiler le programme `myprog.f90` ,
avec production de l'exécutable `myprog` :

```
% ifort -o myprog myprog.f90
```

3) Même objectif avec en plus l'édition de liens
avec la bibliothèque mathématique `libm` :

```
% ifort -o myprog myprog.f90 -lm
```

4) Même objectif que 2) avec en plus l'édition de liens avec la bibliothèque LAPACK fournie dans MKL (et se trouvant dans/softs/intel/v11.1.069/mkl/lib/em64t/) :

```
% ifort -o myptog myprog.f90
  -L/softs/intel/v11.1.069/mkl/lib/em64t -lmkl_lapack
```

5) Création d'une bibliothèque partagée `liboutils.so` à partir de deux fichiers objets

```
% ifort -shared -o liboutils.so fic1.o fic2.o
```

Compilation et/ou édition de liens des programmes séquentiels
Programme reparti sur plusieurs fichiers. Makefile

```
% ifort -o myprog fic1.f fic2.f
```

Inconvénient : Si un fichier source est modifié, cette commande recompile les deux fichiers !

Autre méthode : utilisation d'un fichier Makefile

nom_cible : liste_dependances

```
% ls
 fic1.f fic2.f
 Makefile
% make all
% ls
 fic1.f fic2.f
 Makefile myprog
```

```
% cat Makefile
 all : myprog
 myprog : fic1.o fic2.o
     ifort -o myprog fic1.o fic2.o
 fic1.o : fic1.f
     ifort -c fic1.f
 fic2.o : fic2.f
     ifort -c fic2.f
```

Si un fichier source est modifié, **`make all`** va récompiler seulement ce fichier

Exemples de compilation pour les programmes utilisant OpenMP

• Pour compiler un programme Fortran faisant appel aux directives OpenMP :

% ifort -shared-intel -openmp [options] -o myprog myprog.f90

Remarque : Il faut utiliser le module OpenMP : USE OMP_LIB (avant IMPLICIT NONE)

▪ Pour compiler un programme C faisant appel aux directives OpenMP :

% icc -openmp [options] -o myprog myprog.c

Remarque : Il faut inclure dans le fichier source, au debut, le fichier header omh.h avec l'instruction #include **<omh**.h**>**

▪ Pour compiler un programme C++ faisant appel aux directives OpenMP :

% icpc -openmp [options] -o myprog myprog.cpp

Remarque : Il faut inclure dans le fichier source, au debut, le fichier header omh.h avec l'instruction #include **<omh**.h**>**

Attention A l'exécution, il faut préciser le nombre de threads que l'on veut utiliser

Compilation et/ou édition de liens des programmes parallèles

- **Compilateur Fortran**

mpif77 [options] input_file(s)
ou
mpif90 [options] input_file(s)

- **Compilateur C / C++**

mpicc [options] input_file(s)
ou
mpiCC [options] input_file(s)

où input_files peuvent être :

- des fichier source : – .f .for .ftn .f90 .F .FOR .F90 (pour Fortran et ifort)
  – .C .c .cc .cpp .cxx .c++ (pour C / C++ et icc)

- des fichiers objet : .o

- des bibliothèques statiques : .a

Compilation et/ou édition de liens des programmes parallèles

Remarques sur mpif77, mpif90, mpicc, mpiCC

- Ce ne sont pas de véritables compilateurs, mais juste des interfaces (« wrappers ») avec les compilateurs séquentiels (utilisés pour installer OpenMPI).
- Leur rôle est de
  - rajouter les options nécessaires pour compiler et faire l'édition de programmes MPI
  - faire passer ces options et invoquer le compilateur final (séquentiel)

=> En plus des options des compilateurs séquentiels, les wrappers comportent :

Option	Description	Défaut
-showme	Au lieu d'invoquer le compilateur séquentiel, retourne la commande de compilation qui aurait été exécutée	OFF
-showme:compile	Au lieu d'invoquer le compilateur séquentiel, retourne les options qu'il aurait rajoutés pour la phase de compilation	OFF
-showme:link	Au lieu d'invoquer le compilateur séquentiel, retourne les options qu'il aurait rajoutés pour la phase d'édition de liens	OFF

Compilation et/ou édition de liens des programmes parallèles

Exemple

%source /usr/local/modeles/openmpi-1.4.1-intel-11.1.069

%mpif77 -showme
ifort -I/softs/openmpi-1.4.1-intel/include -L/softs/openmpi-1.4.1-intel/lib -lmpi_f77 -lmpi -lopen-rte -lopen-pal -lrdmacm -libverbs -ldl -lnsl -lutil

%mpif77 -showme:compile
-I/softs/openmpi-1.4.1-intel/include

%mpif77 -showme:link
-L/softs/openmpi-1.4.1-intel/lib -lmpi_f77 -lmpi -lopen-rte -lopen-pal -lrdmacm -libverbs -ldl -lnsl -lutil

=> les deux commandes suivantes sont équivalentes

%mpif77 -c mympiprog.f

%ifort -c mympiprog.f -I/softs/openmpi-1.4.1-intel/include

et aussi les deux suivantes

%mpif77 -o mympiprog mympiprog.o

%ifort -o mympiprog mympiprog.o -L/softs/openmpi-1.4.1-intel/lib -lmpi_f77 -lmpi -lopen-rte -lopen-pal -lrdmacm -libverbs -ldl -lnsl -lutil

Compilation et/ou édition de liens des programmes parallèles

Recommandation

Utiliser les « compilateurs » parallèles (wrappers) au lieu d'utiliser les compilateurs séquentiels en rajoutant manuellement les options d'inclusions et d'éditions de liens vers les bibliothèques OpenMPI

Avantantages

- Les wrappers passent automatiquement les options *-I, -L*, *-l* nécessaires pour les sources utilisant OpemMPI
- Commandes de compilation plus simples
- On peut passer d'une implémentation Open MPI à une autre sans se soucier à refaire les fichiers de Makefile

Compilation et/ou édition de liens des programmes parallèles
Exemples

- Pour compiler un programme Fortran faisant appel à la librairie MPI :

% mpif90 [options] -o mympiprog mympiprog.f90

Remarque : Il faut inclure dans le fichier source, au debut, le fichier header mpif.h avec l'instruction #include **"**mpif.h**"**

- Pour compiler un programme C faisant appel à la librairie MPI :

% mpicc [options] -o mympiprog mympiprog.c

Remarque : Il faut inclure dans le fichier source, au debut, le fichier header mpi.h avec l'instruction #include **<**mpi.h**>**

- Pour compiler un programme C++ faisant appel à la librairie MPI :

% mpiCC [options] -o mympiprog mympiprog.cpp

Remarque : Il faut inclure dans le fichier source, au debut, le fichier header mpi.h avec l'instruction #include **<**mpi++.h**>**

Débogage

Débogage

- **Objectif : éliminer les erreurs dans les programmes →** **débogueur** (le plus souvent)

- **Déboguer un programme consiste :**
  - **à l'arrêter sous certaines conditions pour examiner l'état de la pile d'appels et les valeurs stockées dans les variables**
  - **avec la possibilité de continuer l'exécution du programme pas à pas**

- **L'utilisation d'un débogueur nécessite l'ajout de l'option -g à la compilation**

Débogage : conseils

- **Aérer le code de commentaires**
- **Indenter les lignes du code**
- **Aucune optimisation de code** (l'option -g désactive tout optimisation, sauf si une optimisation est spécifiée de manière explicite-->l'option -debug extended)
- **Afficher les messages de compilation**

Débogage : idb

- **Aérer le code de commentaires**
- **Indenter les lignes du code**
- **Aucune optimisation de code** (-g désactive tout optimisation sauf si une optimisation est spécifiée de manière explicite)
- **Afficher les messages de compilation**