Informatique fondamentale

Domaines

R. Gosswiller

Informatique théorique

Informatique théorique

Qu'est-ce que l'informatique théorique ?

Plusieurs domaines

- Théorie de l'information
- Etude de la complexité
- Théorie des graphes
- Calculabilité
- Théorie des langages

Théorie de l'information

Principe

Etude des modèles de production et de transmission d'information via un canal de communication donné
Etude probabiliste et statistique de l'évolution de l'information dans un système de communication

Applications

Modèles économiques, Encodage de données, ...

Etude de la complexité

Principe

Analyse de la consommation en ressources d'un algorithme

Types de complexités étudiées

Complexité en temps
Complexité en espace

Applications

Algorithmique, ...

Théorie des graphes

Principe

Etude de graphes et modèles pour la résolution mathématique et algorithmique de problèmes fondamentaux

Applications

Réseaux, développpement urbain, ...

Calculabilité

Principe

Vérifier la possibilité d'évaluer une fonction par un algorithme en un temps fini

Démontrer que l'on peut ou non calculer la valeur d'une fonction f en tous points

Applications

Lambda-calcul, Logique combinatoire, Machines de Turing, ...

Théorie des langages

Principe

Etudier la structure, syntaxe et grammaire des langages de programmation

Langages formels

Ensemble de mots formé sur la base d'un alphabet fini
Règles de création, grammaire

Applications

Langages informatiques, ...

Architecture de Von Neumann

Architecture de Von Neumann

?

Il s'agit d'une architecture logicielle permettant de représenter sommairement le comportement d'un ordinateur

Principe

Un ordinateur qui utilise l'architecture de Von Neumann, les programmes et données existent dans une seule forme de mémoire.

Architecture de Von Neumann

Unité de contrôle

- Séquencement des instructions
- Pile
- Gestion du temps
- Envoi de signaux

Opposé à l'architecture de Harvard (séparation mémoire programme/données) utilisée en MCL/DSP

Unité logique et arithmétique

- Exécution des instructions
- Traitement des calculs

Types d'opérations

Arithmétique : $+, -, *, /$ ...
Logique : NON, OU, ET, Compléments, ...
Comparaisons : $==, !=, <, >$...
Binaires : rotations de bits, décalages, ...

Architecture de Von Neumann

Architecture fondamentale
Séparation contrôle/calcul
Instructions et données en mémoire
Un programme = une donnée
Pas de protection de la mémoire

Encodage

Encodage des flottants

Problème

Comment représenter un nombre décimal en mémoire ?

Solution

Décomposition en somme de puissances
Formats différents

Principes

Bases

$14 = 1 * 10^1 + 4 * 10^0 = 1 * 2^3 + 1 * 2^2 + 1 * 2^1 = 1110_b$
Tout nombre peut être exprimé comme une somme de composantes dans sa base d'expression

Décimaux

Il en est de même pour tout nombre décimal
$4,5 = 1 * 2^2 + 1 * 2^{-1}$ Sur 3 bits
$3,375 = 1 * 2^1 + 1 * 2^0 + 1 * 2^{-2} + 1 * 2^{-3}$ Sur 5 bits

Précision

Problème

Plus de précision implique plus de bits

$3,125 = 2^1 + 2^0 + 2^{-3} \rightarrow 5$ bits

$3,123 \approx 2^1 + 2^0 + 2^{-4} + 2^{-5} + 2^{-6} + 2^{-7} \rightarrow 9$ bits

Représenter les flottants

Deux solutions

Virgule fixe

Format avec nombre de bits prédéfini pour chaque partie

Virgule flottante

Utilisation d'une mantisse variable
Gain en précision
Plus lourd, plus difficile à manipuler

Format à virgule fixe

Virgule fixe

Découpage

On associe un nombre de bits donnés à la partie entière
On associe le reste à la partie décimale

Format FP

Notation

$FP_{n,k}$
n bits utilisés
k bits pour la partie décimale
n-k pour la partie entière

Encodage

Somme de puissances de 2, de 2^{-k} à 2^{n-k}

Format FP

Exemple

Encodage en $FP_{8,5}$

-	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}
$3,0125 =$	0	1	1	0	0	1	0	0
$4,257 \approx$	1	0	0	0	1	0	0	0

Inconvénient

Difficulté pour manipuler les puissances négatives

Le changement d'échelle

Propriété

Tout nombre i peut être représenté comme $i * 2^k * 2^{-k}$
Multiplier par 2^k revient à décaler la virgule de k rangs

Calcul

En $FP_{n,k}$, on multiplie i par 2^k
On conserve la partie entière (par troncature), que l'on encode en binaire sur n bits
On divise par 2^{-k} (décalage de k rangs vers la gauche)

Le changement d'échelle

Exemple

Exprimez 3,678 au format $FP_{8,4}$
$3,678 * 2^4 = 58,848$
On encode 58 en binaire : $32 + 16 + 8 + 2 = 00111010$
On redécale de k rangs vers la gauche
Résultat : $0011,1010$

Le calcul de l'erreur

La troncature de $i * 2^k$ implique une perte de données (erreur de précision)
On multiplie la partie décimale obtenue pour obtenir l'erreur
Exemple : $0,848 * 2^{-4} \approx 0,053$ est l'erreur obtenue

Le format BCD

Principe

Binary-coded decimal
Encodage direct (sur 4 ou 8 bits)
Format $BCD_{n,k}$
n symboles à coder, k symboles pour la partie décimale
Format utilisé dans les calculatrices

Exemple

$3,58 = 001101011000$ (format $BCD_{3,2}$)

Format à virgule flottante

La virgule flottante

Problématique

Comment représenter des valeurs de différentes échelles, et différentes précisions, sur le même format ?

Solution

Utiliser un format de représentation dynamique

La notation scientifique

Principe

Toute valeur décimale x peut être exprimée comme la multiplication d'une valeur décimale entre 1 et 10 par une puissance de 10
C'est la forme normalisée d'une valeur, en base décimale

Exemples

$13,37 = 1,337 * 10^{1}$
$456,789 = 4,56789 * 10^{2}$
$0,003456 = 3,456 * 10^{-3}$

La virgule flottante

En binaire

L'expression en notation normalisée est indépendante de la base !
Tout nombre x en binaire peut être exprimé comme : $(+/-)m * 2^e$.

Terminologie

On appelle m la mantisse du flottant
On appelle e son exposant

La virgule flottante

La mantisse

Représenter la valeur à encoder
Mantisse négative : valeur négative

L'exposant

Placement de la virgule dans la valeur finale
Virgule "flottante" : place variable selon la valeur à représenter

Du décimal au binaire

Processus

Encoder en binaire la valeur entière, puis la valeur décimale
Combiner les deux valeurs
Enlever le bit caché (premier bit de la mantisse)

Exemple

$123,375 \rightarrow 123 = 1111011, 0.375 = 011$
$1111011.011 \rightarrow 1.111011011 * 2^6$
$m = 1.111011011, e = 6$
$R = (0)(110)(111011011)$
Il est impossible de retrouver la composition sans connaitre la taille de l'exposant

Erreur relative

Calcul de la précision relative : 2^{-e}

- Simple précision : $\approx 2^{-23}$
- Double précision : $\approx 2^{-52}$

Le format IEEE 754

Principe

Format standard de représentation des flottants
32 ou 64 bits (simple ou double précision)

Répartition

Format	Signe	Mantisse	Exposant	Borne inf.	Borne sup.
32 bits	1	23	8	$1.2 * 10^{-38}$	$3.4 * 10^{38}$
64 bits	1	52	11	$2.2 * 10^{-308}$	$1.8 * 10^{308}$

Du décimal vers IEEE 754

1. Convertir la partie entière en binaire
2. Convertir la partie décimale
3. Combiner les éléments
4. Normaliser la position de l'exposant e
5. Biais de l'exposant : 127 (exposants négatifs)
6. Déterminer le signe s
7. Déterminer l'expression globale : $s * m * 2^e$.

Du décimal vers IEEE 754

Exemple

$100.1875 \rightarrow 100 + 0.1875$
$\rightarrow 0110\ 0100_b + 0.0011_b$
$\rightarrow 1100100.0011_b$
$\rightarrow 1.1001000011_b * 2^6$
$\rightarrow (0)(1000\ 0101)(1001\ 0000\ 1100\ 0000\ 0000\ 000)$

Conclusion

- 5 domaines de l'informatique théorique
- Formats virgule fixe : FP, BCD
- Format virgule flottante
- IEEE 754