روش محاسبه آدرس مستقیم خانه ها درآرایه های سه بعدی ناهموار چگونه است؟ تفاوت این روش با روش مورد استفاده در ماتریس های هموار چیست؟

برای درک کردن روش محاسبه آدرس مستقیم خانه‌ها در آرایه چند بعدی ناهموار، بهتر است اول با روش ساده‌تری که در ماتریس‌های هموار استفاده می‌شود آشنا شویم.

آرایه هموار

برخلاف تصوری که از آرایه‌های چند بعدی داریم، در اکثر زبان های برنامه نویسی این آرایه‌های چند بعدی هموار، به صورت خطی و پیوسته در حافظه ذخیره می‌شوند. این آرایه ها به دو صورت کلی ذخیره می‌شوند:۱. بر اساس سطر ۲. بر اساس ستون.

۱. **بر اساس سطر:** در این رویکرد سطر های آرایه به صورت پشت سر هم در حافظه قرار می‌گیرند. این رویکرد در زبان‌هایی مانند سی و سی پلاس پلاس مورد استفاده است.

فرمول بدست آوردن آدرس عنصر $a[i_1,i_2,\dots,i_n]$ در آرایه n بعدی هموار در این رویکرد:

$$start(a) + (i_1*(d_2*d_3*\dots*d_n) + i_2*(d_3*d_4*\dots*d_n) + \dots + i_{n-1}*d_n + i_n)*sizeOfElement$$

-که در آن start نقطه شروع آرایه در حافظه(آدرس پایه)، d_k سایز بعد k ام و sizeOfElement اندازه اختصاص یافته برای هر عنصر این آرایه است.

۲. **بر اساس ستون:** در این رویکرد ستون‌های آرایه پشت سر هم در حافظه قرار می‌گیرند. این رویکرد در زبان‌هایی مانند متلب و فورتن مورد استفاده است.

فرمول بدست آوردن آدرس عنصر $a[i_1,i_2,\dots,i_n]$ در آرایه n بعدی هموار در این رویکرد:

$$start(a) + (i_n*(d_{n-1}*d_{n-1}*\dots*d_1) + i_{n-1}*(d_{n-2}*d_{n-3}*\dots*d1) + \dots + i_2*d_1 + i_1)*sizeOfElement$$

برای مثال برای آرایه سه بعدی هموار که بر اساس ردیف ها ذخیره شده، ما می‌توانیم از رابطه زیر بدون نیاز به اطلاعات اضافه‌ای استفاده کنیم:

$$Address(A[i][j][k]) = start(a) + ((i*d_{2*}*d_{3)} + (j*d_3) + k)$$

آرایه ناهموار

در آرایه های ناهموار از آنجایی که ردیف ها یا بعدها عددی ثابت نبوده و بلکه متغیر هستند، اوضاع کمی متفاوت می‌شود. این نوع آرایه معمولاً به صورت پیوسته در حافظه ذخیره نمی‌شود و ما برای آدرس دهی به مکانیزم هایی برای دانستن آدرس‌های شروع و یا طول هر زیرآرایه نیاز داریم. در این رویکرد به هر زیر آرایه به صورت جداگانه حافظه تخصیص داده می‌شود و آدرس شروع این زیر آرایه ها(start) نیز ذخیره می‌گردد. پس ما برای دسترسی به هر عنصر، باید ابتدا به آدرس شروع زیر آرایه دسترسی پیدا کنیم.

امیررضا محمدی یگانه

برای مثال آرایه سه بعدی ناهموار A را در نظر می‌گیریم که می‌توان به آن به صورت آرایه ای از زیر آرایه‌های دو بعدی نگاه کرد. این زیر آرایه ها به صورت پیوسته و پشت سر هم در حافظه ذخیره نمی‌شوند و هر کدام آرایه ای از ردیف ها می‌باشند و هر ردیف آرایه ای یک بعدی از عناصر آن ردیف است:

```
A = [[[1, 2], [3]], [[4, 5, 6]]]
```

که می‌توانیم آن را برای صورت آرایه یک بعدی زیر نشان دهیم.

```
a = [1,2,3,4,5,6]
```

همانطور که گفته شد، ما به داده بیشتری برای مشخص کردن آرایه اصلی نیاز داریم. به یک LookUpTable برای هر بعد که آدرس‌های شروع را برای ما نگهداری می‌کند.

LookUpTable1 که آدرس‌های شروع دو زیر آرایه دو بعدی A[0] و A[1] را نگهداری می‌کند:

```
LookUpTable1 = [0,3]
```

و برای هر زیر آرایه دو بعدی، یک LookUpTable دیگر که نقاط شروع هر ردیف در آن زیر آرایه دو بعدی را مشخص میکند:

```
A[0] = [[1,2], [3]]
LookUpTable2[0] = [0, 2]
A[1] = [[4, 5, 6]]
LookUpTable2[1] = [3]
```

فرض کنید می‌خواهیم عدد ۲ را به صورت A[i][j][k] پیدا کنیم. اولاً به آدرس زیرآرایه دو بعدی نیاز داریم که برابر است با:

```
Address_of_2d_subArray = LookUpTable1[0]  #i=0
```

سپس به آدرس ردیف در آرایه دو بعدی i ام نیاز داریم:

```
Address_of_row = LookUpTable2[0][0]  #i=0, j=0
```

و در آخر آدرس عنصر که برابر است با:

```
Address_of_element = Address_of_row + 1*sizeOfElement  #i=0, j=0, k=1
```

البته در زبان‌هایی مانند سی و سی پلاس پلاس می‌توانستیم LookUpTable را به صورت آرایه‌ای از پوینتر ها به نقطه شروع هر زیرآرایه تعریف کنیم.

مقایسه آرایه هموار و ناهموار

همانطور که دیدیم روش محاسبه آدرس مستقیم خانه‌ها در آرایه های هموار و ناهموار به صورت کامل متفاوت است زیرا باتوجه به ثابت بودن طول بعد ها، آرایه های هموار به صورت یه بلوک پیوسته در حافظه ذخیره می‌شوند ولی هر زیر ارایه یک آرایه ناهموار به صورت جداگانه در حافظه ذخیره می‌شود.

محاسبه حافظه در آرایه هموار از یک فرمول ساده بر اساس طول بعدها استفاده می‌شود ولی در آرایه های ناهموار به صورت غیر مستقیم از یک یا چند آرایه کمکی برای پیدا کردن آدرس آن خانه صورت می‌گیرد.

از لحاظ پیچیدگی حافظه‌ای و زمانی کار کردن با آرایه هموار بهتر است زیرا به داده‌های بیشتری برای دسترسی به عناصر نیاز ندارد ولی از آنجایی که طول بعدها در آرایه ناهموار انعطاف پذیر است، در بعضی از مسائل استفاده از آرایه ناهموار مفیدتر می‌باشد.

امیررضا محمدی یگانه