۱.۰ تحلیل اجزای پیچیدگی فضایی

پیچیدگی فضایی از بخش‌های مختلفی تشکیل می‌شود که در جدول ۱ آورده شده است.

جدول ۱: تحلیل اجزای پیچیدگی فضایی

بخش	توضیح	مثال	پیچیدگی
بخش ثابت (C)	حافظه‌ای که همیشه استفاده می‌شود و مستقل از ورودی است.	دستورالعمل‌های برنامه، متغیرهای عمومی	$O(1)$
بخش ورودی ($Sp(I)$)	حافظه‌ای که ورودی‌های برنامه اشغال می‌کنند.	آرایه n عنصری از نوع int $\rightarrow$ $4n$ بایت	$O(n)$
پشته بازگشتی ($Sc(n)$)	حافظه‌ای که برای هر فراخوانی تابع بازگشتی استفاده می‌شود.	فراخوانی‌های بازگشتی در *sum(A, n)*	$O(n)$
حافظه کمکی	حافظه موقتی برای محاسبات یا ساختارهای کمکی.	متغیرهای میانی در محاسبات	وابسته به مسئله

۱.۱.۰ مثال: جمع آرایه با بازگشت

کد زیر مجموع عناصر آرایه را با روش بازگشتی محاسبه می‌کند:

```
int sum(int A[], int n) {
    if (n == 0)
        return 0;
    return A[n-1] + sum(A, n-1);
}
```

تحلیل حافظه:

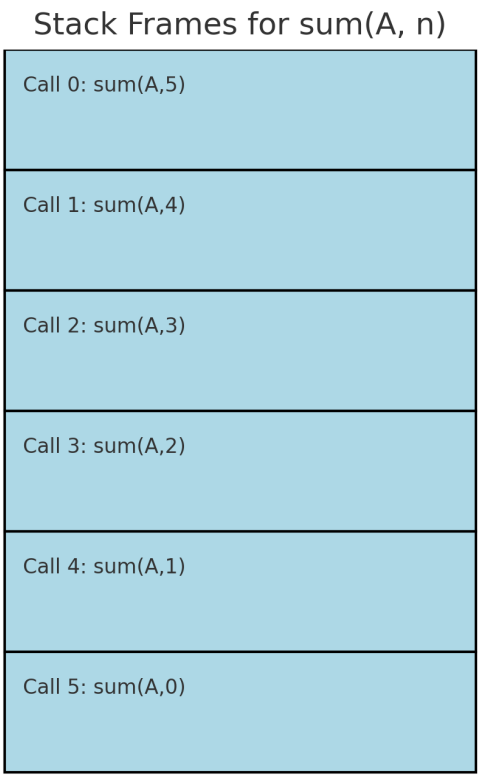
- **بخش ثابت :(C)** فضای لازم برای کد تابع و متغیرهای عمومی، حدود ۲۰۰۰ بایت (مقدار ثابت).
- **بخش ورودی :(Sp(I))** آرایه A با اندازه n، اگر هر عنصر ۴ بایت باشد، $4n$ بایت.
- **پشته بازگشتی :(Sc(n))** هر فراخوانی ۱۶ بایت (آدرس بازگشت + پارامترها)، پس $16n$ بایت.
- **حافظه کمکی:** متغیرهای محلی درون هر فراخوانی ناچیز است.

رابطه کلی:

$$S(P) = C + Sp(I) + Sc(n) = 2000 + 4n + 16n$$

مثال عددی: اگر $n = 1000$:

$$S(P) = 2000 + 4000 + 16000 = 22000 \approx 22$$



شکل ۱: نمایش شماتیک رشد پشته در فراخوانی بازگشتی *sum(A, n)*

۲.۰ روش‌های کاهش مصرف حافظه

برای بهینه‌سازی مصرف حافظه می‌توان از روش‌های مختلفی استفاده کرد که در جدول ۲ خلاصه شده‌اند.

جدول ۲: روش‌های کاهش مصرف حافظه

روش	توضیح	مثال	پیچیدگی فضایی
الگوریتم‌های درجا	داده‌ها در همان محل تغییر می‌کنند و به فضای اضافی زیادی نیاز نیست.	مرتب‌سازی سریع (QuickSort)	$O(\log n)$
ساختارهای داده بهینه	داده به‌صورت فشرده ذخیره می‌شود.	ماتریس تنک (ذخیره فقط مقادیر غیرصفر)، درخت فشرده	بسیار کمتر از حالت عادی
بهینه‌سازی برنامه‌نویسی پویا	نگه‌داری فقط حالت‌های ضروری به‌جای همه حالت‌ها.	ذخیره دو مقدار آخر در فیبوناچی	$O(1)$
الگوریتم‌های جریانی	پردازش داده به‌صورت جریان بدون ذخیره کامل.	HyperLogLog، میانگین‌گیری آنلاین	زیرخطی (Sublin-ear)

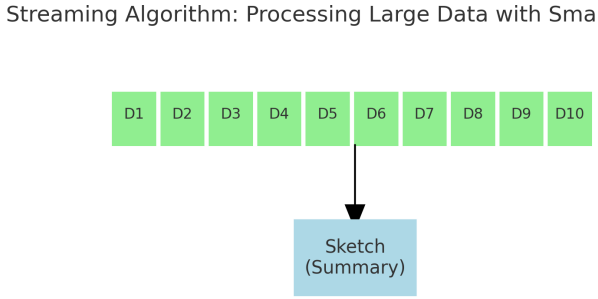
۱.۲.۰ ماتریس تنک چیست؟

ماتریس تنک ماتریسی است که بیشتر خانه‌های آن مقدار صفر دارند و تنها تعداد کمی مقدار غیرصفر دارند. به‌جای ذخیره کل ماتریس، فقط مختصات و مقادیر غیرصفر ذخیره می‌شوند که باعث صرفه‌جویی چشمگیر در حافظه می‌گردد. این روش در مسائلی مانند ذخیره ماتریس‌های بزرگ گراف‌ها یا داده‌های پراکنده در یادگیری ماشین بسیار کاربرد دارد.

۲.۲.۰ توضیح الگوریتم‌های جریانی

الگوریتم‌های جریانی برای زمانی استفاده می‌شوند که داده بسیار بزرگ باشد و امکان ذخیره کل آن وجود نداشته باشد. این الگوریتم‌ها داده را به‌صورت جریان دریافت کرده و خلاصه‌ای کوچک (Sketch) از آن نگه می‌دارند که با استفاده از آن می‌توان تخمین‌های دقیقی انجام داد.

- **مثال:** محاسبه میانگین بدون ذخیره کل داده‌ها (نگه داشتن فقط جمع و تعداد داده‌ها).
- **مثال:** الگوریتم *HyperLogLog* برای تخمین تعداد مقادیر یکتا در دیتاست‌های بسیار بزرگ.



شکل ۲: نمایش شماتیک الگوریتم جریانی: داده به‌صورت جریان وارد شده و خلاصه کوچکی از آن نگهداری می‌شود.