

向量

可扩充向量：算法

$\theta_2 - B_1$

一个人办一县事，要有一省的眼光；办一省事，要有一国之眼光；办一国事，要有世界的眼光

其实“我”不需扩大，宇宙只是一个“我”，只有在我们精神往下陷落时，宇宙与我才分开

邓俊辉

deng@tsinghua.edu.cn

静态空间管理

- ❖ 开辟内部数组 `_elem[]` 并使用一段地址连续的物理空间



- ❖ 若采用静态空间管理策略，容量 `_capacity` 固定，则有明显的不足...
 - 上溢/overflow: `_elem[]` 不足以存放所有元素，尽管此时系统往往仍有足够的空间
 - 下溢/underflow: `_elem[]` 中的元素寥寥无几
- 装填因子/load factor: $\lambda = \text{_size} / \text{_capacity} \ll 50\%$
- ❖ 一般的应用环境中，难以准确预测空间的需求量
- ❖ 可否使得向量可随实际需求动态调整容量，并同时保证高效率？

动态空间管理

- ❖ 蝉：身体经过一段时间的生长，会蜕去原先的外壳，代之以**更大**的新外壳
- ❖ 向量：在即将**上溢**时，**适当扩大**内部数组的容量



(a)



(b)



(c)



(d)



(e)



扩容算法

```
template <typename T> void Vector<T>::expand() { //向量空间不足时扩容
```

```
    if ( _size < _capacity ) return; //无需扩容
```

oldElem



(d)

```
    //assert: _size == _capacity
```

```
    T* oldElem = _elem; //记下原数据区
```

_elem



```
    copyFrom( oldElem, 0, _capacity ); //容量加倍后，复制原数据
```

```
    delete [] oldElem; //释放原空间（得益于封装，不致出现野指针）
```

```
} //为何采用容量加倍策略呢？ 其它策略是否也可行？
```