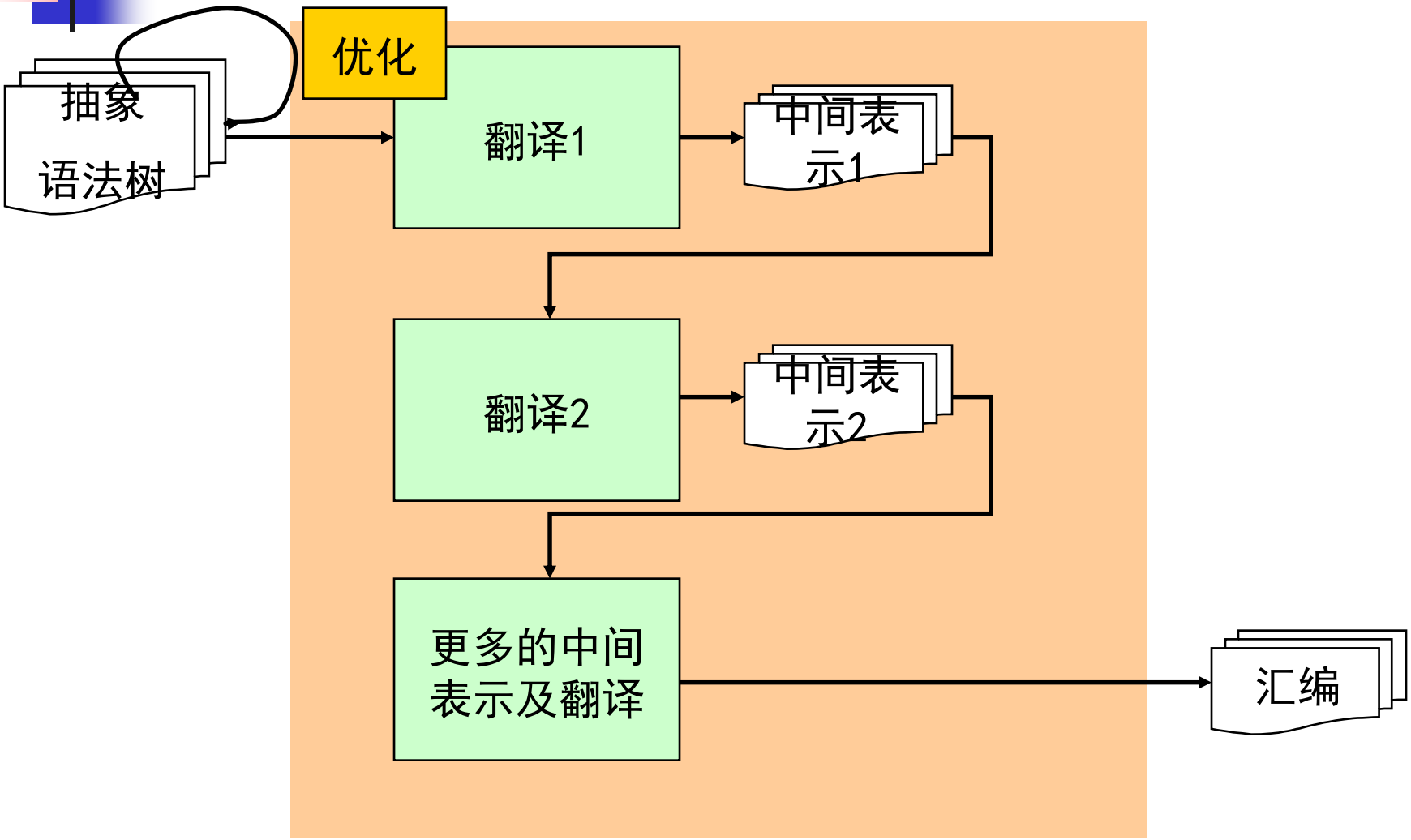
代码优化：前端优化

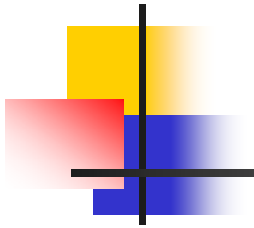
编译原理

华保健

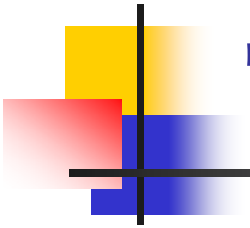
bjhua@ustc.edu.cn

前端优化的地位



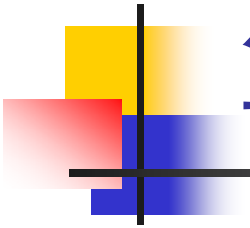


常量折叠



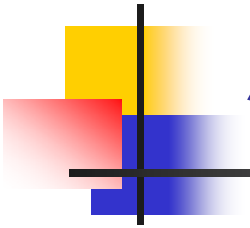
常量折叠

- 基本思想：
  - 在编译期计算表达式的值
    - 例如： a = 3 + 5 ==> a = 8
    - 例如： if (true && false) … ==> if (false)
- 可以在整型、布尔型、浮点型等数据类型上进行



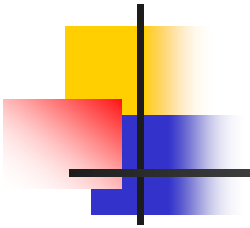
算法

```
// 语法制导的常量折叠算法
const_fold(Exp_t e)
  while (e is still shrinking)
    switch (e->kind)
      case EXP_ADD:
        Exp_t l = e->left;
        Exp_t r = e->right;
        if (l->kind==EXP_NUM && r->kind==EXP_NUM)
          e = Exp_Num_new (l->value + r->value);
        break;
      default:
        break;
```

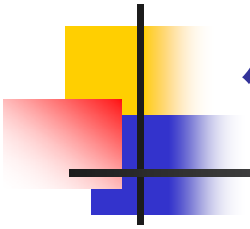


小结

- 容易实现、可以在语法树或者中间表示上进行
- 通常被实现成公共子函数被其它优化调用
- 必须要很小心遵守语言的语义
  - 例如：考虑溢出或异常
    - 例子： 0xffffffff+1 ==> 0 (???)

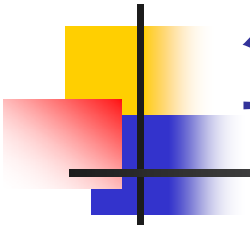


代数化简



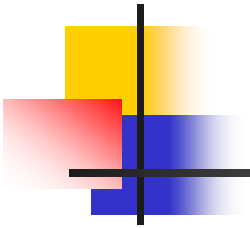
代数化简

- 基本思想：
  - 利用代数系统的性质对程序进行化简
  - 示例：
    - a = 0+b ==> a = b
    - a = 1*b ==> a = b
    - 2*a ==> a + a （强度消弱）
    - 2*a ==> a<<1 （强度消弱）
- 同样必须非常仔细的处理语义
  - 示例： (i-j) + (i-j) ==> i+ i -j -j

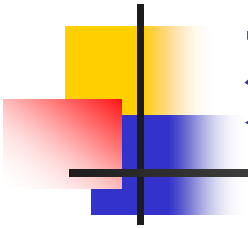


算法

```
// 代数化简的算法
alg_simp(Exp_t e)
  while (e is still shrinking)
    switch (e->kind)
      case EXP_ADD:
        Exp_t l = e->left;
        Exp_t r = e->right;
        if (l->kind==EXP_NUM && l->value==0)
          e = r;
        break;
      case ...:  ...;          // 类似
```

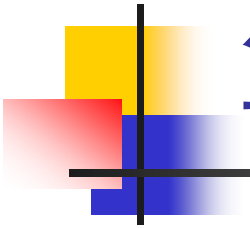


死代码删除



死代码（不可达代码）删除

- 基本思想：
  - 静态移除程序中不可执行的代码
  - 示例：
    - if (false)
      s1;
      else s2;　　==> s2;
- 在控制流图上也可以进行这些优化，但在早期做这些优化可以简化中后端



算法

```
// 不可达代码删除算法
deadcode(Stm_t s)
  while(s is still shrinking)
    switch (s->kind)
      case STM_IF:
        Exp_t e = s->condition;
        if (e->kind==EXP_FALSE)
          s = s->elsee;
        break;
      case ...:  ...;          // 类似
```