Output from Disassembler

```
28: mov   (%esp),%esi
2b: xor   %eax,%eax
2d: xor   %ecx,%ecx
2f:  mov   0x20(%esp),%edi
33: mov   0x1c(%esp),%edx
37: lea   (%edi,%esi,1),%ebp
3a: mov   0x18(%esp),%edi
3e: add   %esi,%edi
40: add   %edx,%esi
42: lea   0x0(%esi),%esi
48: mov   (%edi,%eax,1),%edx
4b: add   $0x1,%ecx
4e: add   (%esi,%eax,1),%edx
51: mov   %edx,0x0(%ebp,%eax,1)
55: add   %ebx,%eax
57: cmp   0x24(%esp),%ecx
5b: jne   48
5d: addl  $0x4,(%esp)
61: cmp   %ebx,(%esp)
64: jne   28
```

j

i

C generated from preprocessing step

Conversion of stack variables

```
bb28:
  eax = 0;
  ecx = 0;
  ebp = stk_esp_x20 + stk_esp;
  edi = stk_esp_x18 + stk_esp;
  esi = stk_esp_x1c + stk_esp;
  bb48:
    edx = *(mem+(edi+eax)/sizeof(int));
    ecx = ecx + 1;
    edx += *(mem+(esi+eax)/sizeof(int));
    *(mem+(ebp+eax)/sizeof(int)) = edx;
    eax += ebx;
    if(stk_esp_x24 != ecx)
        goto bb48;
    stk_esp+=4;
    if(stk_esp!=ebx)
        goto bb28;
```

$(stk\_esp_{initial} + stk\_esp\_x18_{initial}) + ebx*i + 4*j$

$(stk\_esp_{initial} + stk\_esp\_x1c_{initial}) + ebx*i + 4*j$

From profile: ebx >> 4

→ loop interchange (between i loop and j loop) helps locality