CS 389R Project Proposal Presentation

Xiaohui Chen
Department of Computer Science
The University of Texas at Austin

The raising and lowering operator $L_\pm$, $S_\pm$ obey the following properties:

$$L_\pm|l\ m_l\rangle = \hbar\sqrt{l(l+1) - m_l(m_l \pm 1)}|l\ (m_l \pm 1)\rangle \tag{1}$$

$$S_\pm|s\ m_s\rangle = \hbar\sqrt{s(s+1) - m_s(m_s \pm 1)}|s\ (m_s \pm 1)\rangle \tag{2}$$

Since $\vec{j} = \vec{l} + \vec{s}$, $J_\pm$ obeys the following properties:

$$J_\pm = L_\pm + S_\pm \tag{3}$$

$$J_\pm|j\ m_j\rangle = \hbar\sqrt{j(j+1) - m_j(m_j \pm 1)}|s\ (m_j \pm 1)\rangle \tag{4}$$

Therefore a state can be written as

$$|j\ m_j\rangle = \sum_{m_l+m_s=m_j} C^{lsj}_{m_l m_s m_j} |l\ m_l\rangle |s\ m_s\rangle \tag{5}$$

Expample:

We have an electron with angular momentum $l = 1$ and spin $s = \frac{1}{2}$

Therefore $m_l = -1, 0, -1$ and $m_s = \pm\frac{1}{2}$

This means $j = \frac{3}{2}$ and $m_j = -\frac{3}{2}, -\frac{1}{2}, \frac{1}{2}, \frac{3}{2}$

We know that the only possibility of forming $|j\ m_j\rangle = |\frac{3}{2}\ \frac{3}{2}\rangle$ is $|\frac{3}{2}\ \frac{3}{2}\rangle = |1\ 1\rangle|\frac{1}{2}\ \frac{1}{2}\rangle$. This is an instance of equation 5

Applying the lowering operator on the above equation

$$J_-|\tfrac{3}{2}\ \tfrac{3}{2}\rangle = \hbar\sqrt{\tfrac{3}{2}\left(\tfrac{3}{2}+1\right) - \tfrac{3}{2}\left(\tfrac{3}{2}-1\right)}|\tfrac{3}{2}\ \tfrac{1}{2}\rangle = \sqrt{3}\hbar|\tfrac{3}{2}\ \tfrac{1}{2}\rangle$$

$$J_-|1\ \ 1\rangle|\tfrac{1}{2}\ \tfrac{1}{2}\rangle = (L_- + S_-)|1\ \ 1\rangle|\tfrac{1}{2}\ \tfrac{1}{2}\rangle = L_-|1\ \ 1\rangle|\tfrac{1}{2}\ \tfrac{1}{2}\rangle + |1\ \ 1\rangle S_-|\tfrac{1}{2}\ \tfrac{1}{2}\rangle = \hbar\sqrt{1(1+1) - 1(1-1)}|1\ 0\rangle|\tfrac{1}{2}\ \tfrac{1}{2}\rangle + \hbar\sqrt{\tfrac{1}{2}\left(\tfrac{1}{2}+1\right) - \tfrac{1}{2}\left(\tfrac{1}{2}-1\right)}|1\ \ 1\rangle|\tfrac{1}{2}\ \ -\tfrac{1}{2}\rangle = \hbar\sqrt{2}|1\ 0\rangle|\tfrac{1}{2}\ \tfrac{1}{2}\rangle + \hbar|1\ \ 1\rangle|\tfrac{1}{2}\ \ -\tfrac{1}{2}\rangle$$

$$\sqrt{3}\hbar|\tfrac{3}{2}\ \tfrac{1}{2}\rangle = \hbar\sqrt{2}|1\ 0\rangle|\tfrac{1}{2}\ \tfrac{1}{2}\rangle + \hbar|1\ \ 1\rangle|\tfrac{1}{2}\ \ -\tfrac{1}{2}\rangle$$

$$\therefore |\tfrac{3}{2}\ \tfrac{1}{2}\rangle = \sqrt{\tfrac{2}{3}}|1\ 0\rangle|\tfrac{1}{2}\ \tfrac{1}{2}\rangle + \tfrac{1}{\sqrt{3}}|1\ \ 1\rangle|\tfrac{1}{2}\ \ -\tfrac{1}{2}\rangle$$

The coefficients match the Clebsch-Gordan coefficient table

We can apply the lowering operator again on the above equation to get further results

The sum of probabilities is $\frac{2}{3} + \frac{1}{3} = 1$. Therefore the lowering operator method is correct when $l = 1$ and $s = \frac{1}{2}$

The task is to verify the lowering operator method is correct for all allowed l and s

From previous slides, we know that l is integer and s is half integer. Therefore all the real numbers here can by repredented as $\frac{x}{y}$, where x and y are integers

We can represent $\frac{x}{y}$ as a pair of nil-lists (k . a . b). Here a is a list of x nils and b is a list of y nils. The element k is either t or nil, which represents positive number and negative number respetively

e.g. $\frac{2}{3}$ can be represented as ('t '(nil nil) . '(nil nil nil))

Now I only implemened the natural number without the sign

```lisp
;append two nil-lists together
(defun nil-list-plus (x y)
        (if (consp x)
            (cons nil (nil-list-plus (cdr x) y))
            (mapnil y)))

;multiply two nil-lists
(defun nil-list-times (x y)
        (if (consp x)
            (append (mapnil y) (nil-list-times (
             cdr x) y))
            nil))

;add two natural numbers
(defun nat-plus (a b)
        (if (consp a)
            (cons (nil-list-plus (nil-list-times
               (car a) (cdr b))
```

```
                                          (nil-list-times
                                            (cdr a) (car
                                           b)))
                        (nil-list-times (cdr a) (cdr b
                          )))
                nil))

;multiply two natural numbers
(defun nat-times (a b)
        (if (consp a)
                (cons (nil-list-times (car a) (car b
                  ))
                        (nil-list-times (cdr a) (cdr b
                          )))
                nil))

;determine if all elements in the list are nil
(defun true-nilp (x)
        (if (consp x)
```

```
            (and (not (car x))
                 (true-nilp (cdr x)))
            (not x)))

;determine if a pair represent a natural number
(defun true-nil-natpair (x)
        (and (true-nilp (car x))
             (true-nilp (cdr x))))

;determine if a pair represents a natural number
(defun equal-nat (x y)
        (if (consp x)
            (equal (/ (len (car x)) (len (car y)
             ))
                   (/ (len (cdr x)) (len (cdr y)
                     )))
            nil))

(defthm equal-nilp
        (implies (true-nilp x)
                 (equal (mapnil x) x)))
```