1. Given $\psi = C_1 \wedge C_2 \wedge \ldots C_m$ can be the input to a 3SAT problem, where each clause C has 3 literals to choose from $\{x_i, \bar{x}_i \mid 1 \le i \le n\}$. Then to construct a COFFEE problem $\langle G, p, K \rangle$ we have the following specifications. The vertices V of G are $\{v_{c,j} \mid 1 \le c \le m, 1 \le j \le 3\}$ where v_{cj} corresponds to literal j in clause c. We label each vertex $v_{c,j}$ with x_i or $\bar{x}_i$ whichever appears in position j of clause c. For each clause c the edge between each pair of vertices are literals. For each i, an edge between vertices is labeled by x_i and the other by $\bar{x}_i$

The function p maps all vertices to 1. Threshold K is equal to m. ψ is sustainable if and only if the total profit in the COFFEE problem $\langle G, p, K \rangle$ can be at least K.

If we assume that ψ is satisfiable, the assignment A maps to the variables $\{$true, false$\}$. A has to make at least one literal per clause true. Since there's m clauses, this yields exactly $m=K$ vertices, so the total profit is K. Additionally U cannot contain two neighboring vertices in G. If $u, v \in U$ and $(u,v) \in E$, then the edge (u,v) has to be one of the two types mentioned before. u and v cannot correspond to literals in the same clause because only one vertex for each clause was selected, and u and v cannot be x_i and $\bar{x}_i$ for the same i, because A cannot make both the variable and its negation true. Since neither possibilities can be true, U can't have two neighboring vertices. U achieves a total profit of K for the COFFEE problem $\langle G, p, K \rangle$

On the other hand if $U \subseteq V$, $|U| \ge K = m$ exists where no two neighbors in G. U cannot contain neighbors, so it can't contain two vertices from the same clause. So, we must have $|U| = m$ + exactly one vertex from each clause. Suppose: $A(x_i) = $ true if some vertex w/ label x_i is in U and $A(x_i) = $ false if some vertex w/ label $\bar{x}_i$ is in U. U doesn't contain two vertices w/ contradictory labels since U has no neighbors this makes A well-defined which satisfies all clauses by making one literal corresponding to a vertex in U true. A satisfies ψ

If there's a polynomial-time algorithm to output a subset U that maximizes total profit. The algorithm can be easily adapted to a polynomial time algorithm for COFFEE for any $\langle G, p, K \rangle$. For any optimal subset U. True is output for $K \le |U|$ and false otherwise. Since COFFEE is NP-Hard, $P = NP$

2.A. First we must fit the bottom leftmost point like in a Graham scan. Then we sort the rest of the points by angle from that point. We assume that the point we got from the bottom leftmost point is a Ghostbuster. Then take a look at the sorted points by increasing angle, keep track of the difference between # of visited Ghostbusters and Ghosts. When the difference is -1, then stop and connect the point to the bottom, left-most point. The sorting determines the run time which is $O(n \log n)$ - time.

2.B. If we use the algorithm above, we can match one pair of Ghostbusters and ghosts. Each side of the line formed by the pairing the # of Ghostbusters and Ghosts are the same so use the alg. recursively on each side of the line. The worst case is after each iteration and if one side of the line contains no Ghostbusters or ghosts. If that happens then we need $n/2$ total iterations to find pairings which yields a $O(n^2 \lg n)$ time algorithm.

3. Let ψ be the input Boolean formula to a SAT problem and suppose that the set of variables in ψ are $X = \{x_1, x_2 \ldots x_n\}$. Then make a TRIPLE-SAT problem w/ a boolean formula ψ' over a new variable set X' like so:

$$X' = \{x_1, x_2, \ldots x_n, y, z\}$$

$$\psi' = \psi$$

ψ is satisfiable if and only if ψ' has at least 3 satisfying argument. If ψ is satisfiable then any particular assignment can be augmented by adding any of the four possible pairs for $\{y, z\}$ to give at the very least four satisfying assignments overall. But if ψ is not satisfiable then neither is ψ'.

5.A. For all K let $V_1 = 2, S_1 = 1$, and $V_2 = S_2 = 2K$. two item input list
Knapsack capacity = 2K
Since $\frac{V_1}{S_1} = \frac{2}{1} = 2$ and $\frac{V_2}{S_2} = \frac{2K}{2K} = 1$ Alg$_1$ considers the items in order a_1, a_2; including a_1 but excluding a_2, the total value is 2 but the biggest achievable is 2K, which proves Alg$_1$ doesn't guarantee any approximation ratio

5.B. If we let V_{no} be the optimal solution:

$$V_{no} = \sum_{j=1}^{i-1} v_j + \alpha v_i, \text{ where } \alpha \text{ is some fraction and } i \text{ is some item.}$$

Since every Knapsack problem is a valid fractional Knapsack problem, its known that $V_n \leq V_{no}$ then:

$$\max\left(\sum_{j=1}^{i-1} V_j, V_i\right) \leq \max\left(\sum_{j=1}^{i-1} V_j, \alpha V_i\right) \geq \frac{V_{no}}{2}$$

Since one term must be larger than $\frac{V_{no}}{2}$

so it can be looked at as a 2 approximation algorithm.

$$V_0 \leq V_{no} \leq 2\max\left(\sum_{j=1}^{i-1} V_j ; V_i\right)$$