

MODELLI COGNITIVI, COMPETENZA, TRATTABILITA'

MARCELLO FRIXIONE

IIASS «E.R. Caianiello», Vietri sul Mare, Salerno

e.mail: frix@dist.unige.it

1. Competenza e modelli cognitivi

Usualmente, nello studio dei processi cognitivi, le limitazioni di risorse computazionali (tempo di calcolo e spazio di memoria) vengono considerate non pertinenti la sfera della competenza, e di rilevanza esclusiva per lo studio dell'esecuzione. In questo lavoro intendo argomentare, servendomi di considerazioni derivate dalla teoria della complessità computazionale, che vi sono buone ragioni per ritenere che alcuni aspetti inerenti i limiti di risorse possano essere legittimamente considerati pertinenti il dominio di una teoria della competenza.

Le nozioni di *competenza* (*competence*) e di *esecuzione* (*performance*) sono state formulate originariamente da Noam Chomsky in ambito linguistico, con particolare riferimento allo studio della sintassi delle lingue naturali. La distinzione tra competenza ed esecuzione è ben sintetizzata in un passo delle pagine iniziali di *Aspects of the Theory of Syntax*:

La teoria linguistica si occupa in primo luogo di un parlante-ascoltatore ideale, in una comunità linguistica completamente omogenea, il quale conosce perfettamente la sua lingua e non è influenzato da condizioni grammaticalmente irrilevanti quali le limitazioni di memoria, le distrazioni, i cambiamenti di attenzione e di interesse, e gli errori (casuali o caratteristici) nell'applicazione della propria conoscenza della lingua nel corso dell'esecuzione effettiva [...]. Per considerare l'esecuzione linguistica effettiva, dobbiamo considerare l'interazione di vari fattori, e la competenza sottostante del parlante-ascoltatore non è che uno di essi. [...] Introduciamo così una distinzione fondamentale fra *competenza* (la conoscenza che il parlante-ascoltatore ha del proprio linguag-

gio) ed *esecuzione* (l'uso attuale del linguaggio in situazioni concrete). [Chomsky 1965, 4-5].

La competenza linguistica consiste dunque nella conoscenza che un parlante *idealizzato* ha della propria lingua. Un parlante cioè che non sia sottoposto a vincoli di natura extra-linguistica, e di carattere, per così dire, contingente. Tra tali vincoli sono comprese anche le limitazioni di risorse computazionali: Chomsky cita esplicitamente le limitazioni di memoria. Ma considerazioni analoghe possono essere estese senza dubbio anche ai vincoli concernenti i tempi di elaborazione.

Benché formulata originariamente da Chomsky in riferimento al contesto delle ricerche sulla sintassi delle lingue naturali, la nozione di competenza ha tuttavia assunto in seguito una rilevanza più generale per lo studio dei fenomeni cognitivi. Sono state formulate proposte diverse nozioni di competenza sia in relazione all'ambito linguistico, che, più in generale, ad altri compiti cognitivi (cfr. ad esempio [Frixione e Tamburrini 1996] per una rassegna orientata agli aspetti computazionali). Per un punto di vista che colloca la nozione di competenza nella prospettiva dello studio dei processi cognitivi in generale, si veda [Marr 1977].

A David Marr è dovuta anche una caratterizzazione della nozione di competenza che, per quanto non del tutto fedele alla formulazione chomskiana, risulta utile per l'argomento che svilupperò in questo lavoro. Secondo Marr, la spiegazione computazionale di un fenomeno cognitivo può essere formulata a tre livelli differenti, il *livello della teoria computazionale*, il *livello algoritmico* e il *livello dell'implementazione* (si veda ad esempio il primo capitolo di [Marr 1982]). Il livello della teoria computazionale è il più astratto, e concerne la specificazione dei compiti che il possesso di una certa capacità cognitiva permette di assolvere. Tali compiti, a questo livello, sono individuati esclusivamente nei termini dei dati in ingresso e dei risultati prodotti, senza alcun riferimento a procedimenti e meccanismi specifici di esecuzione. In altre parole, in una teoria computazionale un compito cognitivo è specificato nei termini di una pura corrispondenza funzionale fra *input* e *output*. Il li-

vello algoritmico e quello dell'implementazione concernono, a gradi diversi di astrazione, la realizzazione concreta del compito descritto a livello di teoria computazionale. Il livello algoritmico concerne il «come» un certo compito viene effettuato. Concerne cioè le specifiche procedure che lo realizzano, e le strutture di rappresentazione su cui tali procedure sono definite. Il livello dell'implementazione concerne infine le caratteristiche del dispositivo fisico (ad esempio, le strutture del sistema nervoso) che implementa le rappresentazioni e le operazioni proprie del livello algoritmico. La relazione che sussiste fra teoria computazionale e livello algoritmico può essere caratterizzata come la relazione che sussiste tra una funzione e un particolare algoritmo che la calcola. Lo scopo di una teoria computazionale è quello di descrivere una funzione che corrisponda al fenomeno cognitivo che si intende studiare. Tale funzione deve essere effettivamente calcolabile; tuttavia, a livello di teoria computazionale, non viene avanzata alcuna ipotesi circa la natura degli algoritmi e delle procedure che la calcolano.

Secondo Marr, una teoria della competenza si colloca allo stesso livello di una teoria computazionale. Egli afferma ad esempio: «La nozione di teoria della “competenza” per la sintassi dell’inglese di Chomsky [1965] è precisamente quello che io intendo come una teoria computazionale per quel problema» [Marr 1977]. Una teoria della competenza sintattica nel senso di Chomsky sarebbe dunque un caso particolare di teoria computazionale nel senso di Marr.

Il punto di vista di Marr circa il fatto di limitare l'ambito di una teoria della competenza alle sole corrispondenze funzionali fra input e output non sembra coincidere con la posizione di Chomsky. Chomsky ritiene ad esempio che due grammatiche estensionalmente equivalenti (che generano cioè lo stesso linguaggio, e sono quindi indistinguibili nei termini di relazioni input-output) possano non essere egualmente adeguate al fine della rappresentazione della competenza sintattica. La nozione chomskiana di teoria della competenza sembra quindi comprendere aspetti che per Marr ricadrebbero nell’ambito del livello algoritmico di analisi. Tuttavia, ai fini

della presente discussione, è importante rilevare che Marr, rispetto a Chomsky, si limita a restringere il dominio di una teoria della competenza. Si può discutere infatti se una teoria della competenza secondo Chomsky abbia a che fare *solo* con la funzione computata (e probabilmente, come si è detto, non è così). Ma le considerazioni inerenti la funzione computata sono certamente rilevanti anche per una teoria della competenza nel senso di Chomsky. Quindi, tutti gli aspetti della competenza intesa nel senso di Marr sono pertinenti anche la nozione chomskiana di competenza. Per il seguito di questo lavoro, quindi, l'adozione di un'accezione di competenza *a la* Marr non comporta alcuna perdita di generalità: se un certo aspetto è rilevante per la nozione marriana di teoria della competenza, allora quell'aspetto deve essere rilevante, a maggior ragione, per la nozione chomskiana. Dal punto di vista che qui ci interessa, la definizione di Marr ha il vantaggio di isolare alcuni aspetti ben identificabili dal punto di vista formale: una teoria della competenza per un dato compito cognitivo deve avere a che fare (almeno) con la *funzione* (nel senso matematico) che corrisponde a quel compito.

2. Due compiti difficili

Torniamo ora al tema dei limiti di risorse computazionali. Come dicevo, tradizionalmente tali aspetti sono stati considerati irrilevanti per il livello della competenza, e sono stati relegati all'ambito dell'esecuzione. Vi sono certamente casi in cui i limiti di risorse computazionali possono risultare rilevanti soltanto a livello di teoria dell'esecuzione. Per altri aspetti tuttavia ritengo vi siano buone ragioni per prendere in considerazione l'ipotesi che le limitazioni di risorse possano essere pertinenti anche a livello di studio della competenza.

Prendiamo in considerazione due esempi di compiti «difficili» dal punto di vista delle risorse computazionali, problemi che possono essere considerati, in un certo senso, paradigmatici dei due casi sopra citati. Il primo compito, chiamiamolo (*a*), consiste nel

memorizzare e ripetere sul momento una sequenza di numeri naturali, compresi, poniamo, tra 1 e 1000. Se i numeri da memorizzare sono abbastanza (ad esempio cento, ma ne bastano molti meno) il compito diventa di una difficoltà proibitiva per la totalità degli esseri umani. La capacità della nostra memoria a breve termine è infatti estremamente più limitata: lo psicologo cognitivo G.A. Miller [1956] ha mostrato, sulla base di una serie di esperimenti relativi a compiti diversi, che essa ha una capacità di circa sette elementi distinti (Miller parla di «magico numero sette più o meno due»).

Il secondo compito, chiamiamolo (*b*), è il cosiddetto problema *SAT*, vale a dire il classico problema di decidere se una formula del linguaggio della logica proposizionale è soddisfacibile, cioè di stabilire che non si tratti di una contraddizione. Sia F una formula proposizionale, e siano p_1, \dots, p_n le n lettere proposizionali distinte che vi compaiono. Per decidere la soddisfacibilità di F , supponiamo di usare un algoritmo basato sulle tavole di verità. Se F ha n lettere proposizionali distinte, il numero delle righe della tavola di verità di F sarà 2^n . Nel caso peggiore, per stabilire se F è soddisfacibile, si dovranno considerare tutte le 2^n righe della tavola. Quindi, nel caso peggiore, la durata del calcolo è di un ordine di grandezza esponenziale rispetto al numero di lettere proposizionali diverse nella formula. Se F ha 30 lettere proposizionali distinte, nel caso peggiore si dovranno considerare più di un miliardo di righe diverse. Questo certamente eccede le capacità di un soggetto cognitivo reale.

Abbiamo a che fare dunque con due compiti che diventano rapidamente proibitivi per qualunque soggetto umano. Il compito (*a*), tuttavia, è in qualche modo «difficile» dati i limiti contingenti del dispositivo che lo esegue (in questo caso, la capacità della memoria umana a breve termine). Il compito di per sé non è *intrinsecamente* difficile. E' banale progettare una macchina che sia in grado di eseguirlo in maniera efficiente. Basterebbe poi riuscire a manipolare la memoria a breve termine di un essere umano in modo da estenderne la capacità ad un numero arbitrario di elementi, e questo diventerebbe un compito banale anche per lui. Il com-

pito (*b*) invece è difficile per qualunque dispositivo. Ogni volta che si aumenta di 1 il numero delle lettere proposizionali distinte della formula F , le risorse di calcolo devono raddoppiare. Ovviamente, anche nel caso del compito (*a*), per qualunque macchina reale, esisterà sempre una sequenza di numeri troppo lunga per essere memorizzata. Ma basta ampliare di poco le capacità della macchina per superare questo limite. Nel caso di un compito come (*b*) invece, quando la difficoltà del problema aumenta di poco, le risorse per risolverlo devono aumentare di molto. Ben presto, per dati di dimensioni apparentemente «ragionevoli», si arriva a richiedere risorse di calcolo del tutto irrealizzabili. Ad esempio, anche supponendo di disporre di dispositivi dotati della massima velocità di calcolo fisicamente ipotizzabile, si raggiungono rapidamente tempi di elaborazione che supererebbero la vita dell'universo. In questo senso, usualmente vengono considerati *computazionalmente intrattabili* quei compiti che richiedono risorse di calcolo (tempo o spazio di memoria) che crescono esponenzialmente rispetto alla lunghezza dell'input (misurata in maniera opportuna; ad esempio, nel caso di *SAT*, la lunghezza dell'input è data dal numero di lettere proposizionali distinte della formula F). Vengono invece considerati *computazionalmente trattabili* quei compiti che richiedono risorse che crescono al più in modo polinomiale.

Si potrebbe obiettare che considerazioni come quelle relative alla durata della vita dell'universo costituiscano fatti puramente contingenti, che nulla hanno a che fare con limitazioni di principio (ad esempio, è banale definire compiti che siano computazionalmente trattabili in base alla caratterizzazione data sopra, che tuttavia, anche per input molto piccoli, richiedono risorse di calcolo proibitive). Tuttavia il punto principale consiste piuttosto nel fatto che per un compito come (*b*) il rapporto tra il tempo di calcolo richiesto e la lunghezza dell'input cresce con una rapidità tale da far ritenere che, in un certo senso, il compito nella sua generalità sia *in linea di principio* non realizzabile. In altri termini, ipotizzare un dispositivo idealizzato che esegua un compito come (*b*) sembra ri-

chiedere un grado di idealizzazione maggiore che non immaginare un dispositivo idealizzato che esegua un compito come (a).

Si potrebbe obiettare inoltre che un algoritmo basato sulle tavole di verità potrebbe essere un cattivo algoritmo per decidere la soddisfacibilità di una formula proposizionale, e che, se si scegliesse un algoritmo migliore, le cose potrebbero andare diversamente. In un certo senso, le cose stanno effettivamente in questo modo, in quanto un algoritmo basato sulle tavole di verità è particolarmente inefficiente, ed esistono altri algoritmi molto migliori per svolgere questo compito. Tuttavia, tutti gli algoritmi noti, nei casi peggiori, richiedono tempi di calcolo dell'ordine dei 2^n passi di calcolo, dove n è una misura appropriata della lunghezza della formula.

3. Classi di complessità computazionale

Il punto è che presumibilmente un compito come (b) è intrinsecamente difficile - la sua difficoltà dipende cioè dal compito in sé (ossia, dalla funzione) e non dall'algoritmo scelto per risolverlo. Per chiarire questa affermazione è opportuno introdurre alcuni concetti della teoria della complessità computazionale (una utile introduzione alla teoria della complessità computazionale è [Stockmeyer 1987]; per un trattamento approfondito si veda ad esempio [van Leeuwen 1990]). Sia P la classe dei problemi di decisione (quei problemi cioè che ammettono una risposta di tipo «sì o no») che sono risolvibili in tempo polinomiale da una macchina di Turing deterministica (d'ora in avanti, MTD). Una MTD è una Macchina di Turing usuale, in cui ogni passo di calcolo è determinato univocamente dalla configurazione della macchina (cioè dalla coppia stato della macchina-simbolo osservato sul nastro). Sia invece NP la classe dei problemi di decisione risolvibili in tempo polinomiale da una macchina di Turing *non* deterministica (in breve, MTN). Una MTN è una macchina di Turing nella cui tavola per ogni configurazione può esistere più di una quintupla, ossia più di un'istruzione che può essere eseguita. Questo fa sì che, a partire da

un certo input, possa esistere più di una computazione possibile: dato l'input e una configurazione iniziale, ci sarà un albero di possibili computazioni, dove ogni biforcazione corrisponde alla scelta non deterministica tra due istruzioni possibili. Un problema di decisione è un problema NP se esiste una MTN tale che, per ogni input, se la risposta è positiva esiste almeno una computazione di lunghezza al più polinomiale che la produce.

Si può comprendere meglio il senso di questa definizione mostrando in maniera intuitiva che *SAT* è un problema NP. Sia MT_{SAT} una MTN che, presa in input una formula proposizionale F opportunamente codificata, decida se F è soddisfacibile procedendo nel modo seguente. Per ogni lettera proposizionale p_i in F , MT_{SAT} assegni non deterministicamente un valore di verità, vero o falso, a p_i . MT_{SAT} produrrà quindi una possibile computazione per ogni possibile assegnazione di valori di verità alle lettere proposizionali di F (cioè, per ogni riga della tavola di verità di F). Per ogni assegnazione, MT_{SAT} verificherà quindi se questa assegnazione di valori di verità soddisfa F . Se F è soddisfacibile, deve esistere almeno una di queste computazioni che produce una risposta positiva. Poiché decidere se una data assegnazione di valori di verità soddisfa o meno una formula proposizionale richiede un tempo di lunghezza polinomiale (anzi, addirittura lineare) rispetto alla lunghezza della formula stessa, tutte le possibili computazioni di MT_{SAT} saranno di lunghezza polinomiale rispetto alla lunghezza di F . Quindi, se F è soddisfacibile, esisterà almeno una computazione di MT_{SAT} di lunghezza polinomiale che lo determina (in particolare, saranno tali tutte quelle computazioni in cui MT_{SAT} «indovina» le assegnazioni giuste, cioè quelle che soddisfano F).

E' ovvio constatare che tutti i problemi in P sono anche in NP, ossia che $P \subseteq NP$ in quanto, banalmente, ogni MTD è anche un caso limite di MTN. Resta tuttavia aperto il problema di stabilire se valga anche l'inclusione inversa, il che comporterebbe che $P = NP$. Anche se per il momento non è possibile disporre di alcuna dimostrazione al proposito, esistono tuttavia buoni motivi per cre-

dere che le cose non stiano in questo modo. Vediamo le ragioni che motivano questa convinzione.

Definiamo la classe dei problemi NP-*completi* nel modo seguente. Diremo che un problema p è NP-completo se è compreso in NP, e se gode inoltre della proprietà per cui ogni altro problema in NP è riducibile (in tempo polinomiale) a p , nel senso che, intuitivamente, se si disponesse di un algoritmo «efficiente» per p , allora si potrebbe ottenere (in tempo polinomiale, appunto) un algoritmo efficiente per qualsiasi problema in NP.

Intuitivamente quindi i problemi NP-completi sono i problemi «più difficili» in NP. Si dimostra che *SAT* appartiene alla classe dei problemi NP-completi (e, per ragioni storiche, può essere considerato un esempio «paradigmatico» di questa classe). Ora, non è mai stato trovato nessun algoritmo che risolvesse in tempo polinomiale alcun problema NP-completo. Poiché dalla definizione di NP-completezza segue banalmente che tutti i problemi NP-completi sono riducibili gli uni agli altri, basterebbe individuare un algoritmo trattabile per un qualsiasi problema NP-completo per disporre di un algoritmo trattabile per *tutti* i problemi NP-completi (nonché per tutti i problemi NP). Dato il grande sforzo destinato a individuare algoritmi trattabili per problemi NP-completi, e data la vanità di tale sforzo, si assume usualmente che valga

(*) $P \neq NP$,

sebbene si tratti di una congettura di cui non è disponibile per il momento alcuna dimostrazione.

Qualora tale congettura risultasse vera, esisterebbero dei problemi in NP (e i problemi NP-completi sarebbero tra questi) che sono per loro stessa natura non trattabili computazionalmente, per i quali non esiste cioè alcun algoritmo che li risolva entro limiti di risorse ragionevoli.

E' importante notare che qui si parla di *problemi*, e non di singoli algoritmi, né di modelli di dispositivi di calcolo specifici. Si assume infatti che i risultati sopra ricordati siano indipendenti dal

modo in cui il processo di calcolo viene caratterizzato. Questa assunzione si basa sulla constatazione che tutte le classi di macchine «ragionevoli» che sino ad ora sono state studiate si sono rivelate equivalenti da questo punto di vista, nel senso che si possono simulare reciprocamente con un sovraccarico al più di ordine polinomiale. E' quindi verosimile ipotizzare che non verranno scoperti modelli di calcolo ragionevoli che la falsifichino. (Qui ovviamente andrebbe precisato cosa si intende per «ragionevoli» - in questa sede non è possibile approfondire l'argomento, ma si veda ad esempio [van Leeuwen 1990, cap. 1]). La formulazione precisa di questa assunzione viene detta *Tesi dell'invarianza*. Intuitivamente, la Tesi dell'invarianza si può enunciare dicendo che se un problema non è computazionalmente trattabile per una macchina di Turing (deterministica), allora non è computazionalmente trattabile per alcun dispositivo di calcolo ragionevole. Dal punto di vista epistemologico, la Tesi dell'invarianza ha uno statuto simile a quello della Tesi di Church: non si tratta di una congettura, nel senso che non si tratta di un enunciato passibile di essere un giorno dimostrato e di diventare quindi un teorema; si tratta di piuttosto di un'ipotesi che in linea di principio potrebbe risultare falsificata, ma per la quale tuttavia è stata accumulata un grande mole di evidenza positiva.

Prima di procedere, sono opportune due precisazioni. In primo luogo, nelle pagine precedenti le classi P e NP sono state definite in maniera tale da includere esclusivamente problemi di decisione (cioè problemi che ammettono esclusivamente risposte di tipo «sì o no»). Tuttavia, è possibile generalizzare tali definizioni in modo da comprendere anche problemi di forma diversa (ad esempio, problemi di ricerca, o, in generale, funzioni computabili di tipo qualunque), e tutto quanto si è detto vale anche in questo caso più generale. In secondo luogo, l'intrattabilità computazionale non è limitata ai problemi NP-completi. La classe dei problemi NP-completi riveste particolare importanza in quanto include problemi di grande rilevanza sia teorica che pratica. Tuttavia, esistono altre classi di problemi delle quali si è congetturato che non siano tratta-

bili computazionalmente (e più difficili di NP). Esistono inoltre problemi che risultano essere non trattabili computazionalmente a prescindere dalla verità di alcuna congettura, problemi cioè per i quali è stato possibile *dimostrare* che non possono essere risolti in tempo polinomiale da alcuna MTD (e che quindi, in base alla Tesi dell'invarianza, si è *dimostrato* non essere trattabili). (Su tutti questi aspetti si veda ad esempio [van Leeuwen 1990, cap. 2]).

4. Competenza trattabile

Dunque, in base a quanto detto, le proprietà inerenti la trattabilità computazionale di un compito non sono relative a un modo specifico per risolvere un problema (uno specifico algoritmo, o un tipo di macchina) ma inerenti il problema stesso. Ossia, in altri termini, le proprietà computazionali si riferiscono alla *funzione* calcolata e non ai metodi per calcolarla. Tornando quindi al contesto dei processi cognitivi, le considerazioni relative alla trattabilità computazionale si possono legittimamente considerare come pertinenti il *livello delle teorie computazionali* nel senso di Marr, e quindi l'ambito di una *teoria della competenza*.

Da queste considerazioni emerge una visione più restrittiva della nozione di competenza, che consente di imporre maggiori vincoli sulla nozione di soggetto competente idealizzato. Tale nozione di «competenza trattabile» rimane infatti ancora una idealizzazione rispetto alle capacità dei soggetti cognitivi reali. In questo senso, che sia lecito considerare gli aspetti di complessità computazionale come pertinenti il livello della competenza emerge anche dalle considerazioni seguenti. Dato un certo compito cognitivo C , si supponga di individuare una funzione che lo modella (chiamiamola f_C) che sia computazionalmente trattabile. Ora, f_C costituisce certamente ancora una idealizzazione rispetto al compito C quale è effettuato dagli esseri umani. Ad esempio, f_C non tiene assolutamente conto delle specifiche limitazioni di risorse proprie dei soggetti umani in carne e ossa: essa può ammettere input di dimensioni arbitrarie, o computazioni in cui la memoria richiesta eccede quella

di qualunque essere umano, o la cui lunghezza supera quella di qualsiasi esistenza umana. Quindi, imporre un vincolo di trattabilità computazionale su f_C non ha a che fare con i vincoli propri di una teoria dell'esecuzione.

Una possibile obiezione alla tesi dell'importanza cognitiva dei risultati della teoria della complessità computazionale, e della loro rilevanza per la formulazione di teorie della competenza, concerne il fatto che i risultati della teoria della complessità sono relativi al *caso peggiore*: l'appartenenza di un problema a una data classe di complessità viene determinato sulla base delle istanze del problema che richiedono risorse computazionali più onerose. Si potrebbe sostenere che sarebbe cognitivamente più rilevante fare riferimento al caso medio anziché al caso peggiore. Due considerazioni sono opportune a questo proposito. In primo luogo, lo studio della complessità computazionale media dei problemi è per il momento meno sviluppata, per ragioni tecniche, dello studio del caso peggiore. Tuttavia, le proprietà computazionali di un problema nel caso medio sono comunque proprietà della *funzione* computata, e pertanto la loro rilevanza cognitiva è comunque in accordo con la mia tesi principale, della rilevanza cioè di alcuni aspetti relativi ai limiti di risorse computazionali per una teoria della competenza. In secondo luogo, l'assunto del maggiore interesse cognitivo del caso medio rispetto al caso peggiore è legato alla constatazione che i soggetti cognitivi reali nell'affrontare compiti computazionalmente intrattabili impiegano strategie che *approssimano* i risultati del compito definito nella sua globalità, «scartando», per così dire, i più onerosi. Ora, lo studio di soluzioni approssimate per problemi intrattabili è una delle maggiori aree di ricerca in teoria della complessità. Data una funzione f computazionalmente intrattabile, trovare una approssimazione trattabile per f significa individuare una funzione f' che sia computazionalmente trattabile, e il cui comportamento stia in una relazione opportuna con il comportamento di f . Assumere che lo studio del caso peggiore sia cognitivamente rilevante per una teoria della competenza comporta che lo studio di una tale f' sia pertinente al livello della competenza, che cioè lo

studio delle soluzioni approssimate impiegate dai soggetti cognitivi non sia relegabile interamente all'ambito dell'esecuzione (per una discussione di questo tipo di obiezioni si veda anche [Levesque 1988, 364-365]).

5. Vincoli della competenza, vincoli dell'esecuzione

In generale, una motivazione per prendere in considerazione una nozione di competenza «trattabile» scaturisce da considerazioni di tipo metodologico e di tipo filosofico, relative alla plausibilità e alla praticabilità di una visione computazionale della mente. Un modello computazionale di un determinato compito cognitivo che non tenga conto dei vincoli di complessità computazionale può dirci molto poco su come quel compito è effettivamente realizzato dalla mente, e ha quindi un limitato valore esplicativo. Utilizzando le parole di Hector Levesque, un modello del genere «tells us *nothing* about how the cognitive activity is physically possible» [Levesque 1988, 359]. Disporre di un modello della competenza basato su funzioni computabili, ma non trattabili computazionalmente comporta che può certamente esistere una macchina (di Turing) che esegue quel compito, ma ciò non contribuirebbe a fornire una spiegazione adeguata di come quel processo viene eseguito da un essere umano. Da un punto di vista specifico di filosofia della mente, un modello della competenza che non tenga conto dei vincoli di trattabilità computazionale ha poco valore a sostegno di una visione computazionale della mente, in quanto sarebbe del tutto legittimo sostenere che gli esseri umani svolgono quel compito in maniera sostanzialmente diversa da come lo esegue la macchina. Modelli del genere non sono di alcun sostegno contro l'affermazione che gli esseri umani sanno fare cose che nessuna macchina «realistica» è in grado di fare.

D'altro canto, il punto che intendo sottolineare non consiste tanto nel fatto che una teoria della competenza *debba* tenere conto delle limitazioni di risorse computazionali. Il punto è piuttosto che *può* farlo, e che questo può consentire di elaborare modelli della

competenza più dettagliati e più adeguati. Consente ad esempio di non relegare a livello dell'esecuzione tutta una serie di elementi cruciali. Ho già fatto cenno al problema dell'approssimazione, vedremo nel seguito come sia stato suggerito che la teoria della complessità computazionale potrebbe consentire di dar conto del *perché* della sistematicità di certi «errori» e di certi comportamenti devianti rispetto ai canoni della razionalità normativa. Un altro aspetto può consistere nella possibilità di distinguere tra aspetti diversi inerenti le limitazioni di risorse.

Infatti, come già si è detto (§ 2), non tutti i vincoli relativi alle limitazioni di risorse possono essere ritenuti pertinenti una teoria della competenza «trattabile». Anzi, questo approccio consente di distinguere due livelli diversi: (a) vincoli inerenti il compito cognitivo in quanto tale (e pertinenti quindi la teoria della competenza) e (b) vincoli inerenti aspetti specifici dell'architettura cognitiva (e pertinenti quindi il piano dell'esecuzione, o addirittura il livello implementativo di Marr). Un caso del secondo tipo è offerto dal già citato *magico numero 7 ± 2* di Miller. Alcuni degli elementi (anche se non tutti) avanzati dai connessionisti a favore dei modelli cognitivi distribuiti in parallelo contro i modelli cognitivi classici si collocano al livello (b). Un esempio è costituito dal cosiddetto *limite dei cento passi* [Feldman e Ballard 1982]. Il limite dei cento passi ha a che fare col fatto che i neuroni sono lenti rispetto ai calcolatori digitali: la durata di una singola «operazione» in un neurone viene stimata nell'ordine della decina di millisecondi. Perciò, tutti i fenomeni cognitivi di durata compresa in un secondo devono richiedere un calcolo la cui lunghezza sia dell'ordine di circa cento passi. Questo non può essere certamente ottenuto se non mediante un dispositivo di calcolo di tipo parallelo. Per tali ragioni, un modello computazionale rigidamente sequenziale come quello basato sull'architettura di von Neumann non può essere adeguato a spiegare come viene elaborata l'informazione nel sistema nervoso. E' evidente che considerazioni di questo tipo nulla hanno a che fare con le proprietà computazionali del problema, ma esclusivamente con il modo in cui è organizzata l'architettura cognitiva e con le

proprietà biofisiche dei suoi componenti. In questo senso, il vincolo dell'implementazione risulta pertinente il livello della implementazione nella gerarchia di Marr.

Una posizione che, pur non menzionando esplicitamente la nozione di competenza, è sulla stessa linea di quella qui sostenuta è formulata nel già citato lavoro di Hector Levesque [1988], che è interessato in maniera specifica ai modelli logici del ragionamento. Secondo Levesque, studiare i compiti cognitivi dal punto di vista delle risorse computazionali con gli strumenti della teoria della complessità significa considerare tali compiti come compiti di elaborazione dell'informazione, cioè come associazioni funzionali da insiemi di input a insiemi di output [Levesque 1988, 360]. Sono evidenti le analogie tra questa impostazione e il livello della teoria computazionale secondo la formulazione di Marr.

Un approccio basato sulla teoria della complessità consentirebbe di trattare a livello, per così dire, di teoria computazionale aspetti tradizionalmente ritenuti pertinenti la sfera dell'esecuzione: “[...] memory limitations, attention lapses, or anything else that may be viewed as less than ideal. This is all part of the subject matter, and needs to be modelled deliberately as part of an IP [Information Processing] task” (*ibid.*, 364). Sono interessanti le conseguenze che, secondo Levesque, tale impostazione comporterebbe per lo studio del ragionamento di senso comune: «My thesis is as follows. The deviations from classical logic that will be necessary to ensure the tractability of reasoning stand in very close correspondence to the deviations from logic that we would have to make anyway to be psychologically realistic. If we look at the kinds of mistakes people make, the kinds of problems people run into, and the corners that are cut to get around them, we will find modifications to classical logic that ensure the computational tractability of the associated thinking» (*ibid.*, 369-370).

Riferimenti bibliografici

Chomsky, N.

- 1965 *Aspects of the Theory of Syntax*, Cambridge (Mass.), MIT Press, tr. it. in N. Chomsky, *La grammatica generativa trasformazionale. Saggi linguistici*, vol. II, Torino, Boringhieri, 1970.
- Feldman, J.A. e Ballard, F.H.
 1982 Connectionist models and their properties, *Cognitive Science*, 6, 205-254.
- Frixione, M. e Tamburrini, G.
 1996 Conflitti di competenza, *Sistemi intelligenti*, 8(3), 375-402.
- van Leeuwen, J. (a cura di)
 1990 *Handbook of Theoretical Computer Science. Volume A. Algorithms and Complexity*, Amsterdam, Elsevier.
- Levesque, H.J
 1988 Logic and the complexity of reasoning, *Journal of Philosophical Logic*, 17(4), 355-389.
- Marr, D.
 1977 Artificial intelligence: a personal view, *Artificial Intelligence*, 9, 37-48.
 1982 *Vision*, New York, Freeman.
- Miller, G.A.
 1956 The magical number seven, plus or minus two: Some limits in our capacity for processing information, *Psychological Review*, 63, 81-97.
- Stockmeyer, L.
 1987 Classifying the computational complexity of problems, in *Journal of Symbolic Logic*, 52(1), 1-43.