1 INTRODUZIONE

CONTESTO APPLICATIVO

Il mio lavoro di tesi riguarda lo studio di alcuni modelli non convenzionali di calcolo, proposti dalla biologia (**SPLICING SYSTEM**).
Le teoria presentata in questa tesi riguarda lo studio di un particolare comportamento ricombinante operato su molecole di DNA, RNA, detto *splicing*.

MOTIVAZIONI

Lo scopo è di creare prodotti software che riescano a simulare lo splicing offrendo una buona analisi dei dati ottenuti.
Per affrontare il problema dello splicing, è stato fatto uno studio approfondito della teoria **dei linguaggi formali** usando come mezzi a disposizione **teoremi e definizioni** forniti da Tom Head che, nel 1987 ha introdotto i sistemi splicing come astrazione dello splicing biologico.
Queste informazioni hanno permesso di realizzare prodotti software che simulano lo splicing.
In un Sistema Splicing, un linguaggio può essere associato a:

1. Un insieme costituito da molecole di DNA.
2. Un insieme costituito da gruppi di enzimi che permettono di legare coppie di molecole di DNA.

In generale, questi tipi di linguaggi consistono in stringhe di simboli che rappresentano le strutture primarie delle molecole di DNA che possono essere prodotte a partire da molecole di DNA originali grazie a gruppi di enzimi. Nonostante molti risultati teorici dati in letteratura, altri restano

aperti.

L'obiettivo principale è stato di analizzare e modellare i *sistemi splicing circolari* proprio perché avere uno strumento software che generi le parole per splicing può dare la possibilità di analizzare le proprietà e scoprire caratteristiche comuni alle stringhe generate.

OBIETTIVI E RISULTATI RAGGIUNTI

La modellazione è stata fatta attraverso il linguaggio di programmazione *Java,* utilizzando come ambiente di sviluppo *Eclipse*.

Un *sistema splicing circolare* è una tripla (A, I, R) dove A è una alfabeto finito, I è il linguaggio circolare finito e R è l'insieme finito delle regole splicing.

A partire dai sistemi *splicing circolari (1-3)*[3], è stato implementato un sistema in cui si consente l'uso di qualsiasi tipo di regola (*splicing circolare generale).*

L'implementazione ha richiesto l'uso di ArrayList, e mappa come strutture dati di supporto.

Per quanto riguarda le regole è stato usato semplicemente un array list, dove ad ogni posizione viene inserito un array composto da quattro elementi che a sua volta contengono i siti di restrizione. La gestione del linguaggio circolare iniziale è stata fatta utilizzando un pacchetto sviluppato in[].

Il cuore del progetto è la generazione, per un k fissato dall'utente, del linguaggio circolare generato dal sistema dato in input, usando la definizione di linguaggio splicing data in letteratura. Esso non è nient'altro che l'insieme di parole ottenute a partire da I per l'applicazione iterata delle regole a I e poi alle parole (circolari) via via ottenute.

Ad ogni iterazione si è provato di evitare di calcolare un'operazione splicing già calcolata nei passi precedenti. Come struttura dati è stata usata una mappa e le motivazioni sono fornite nel documento presente.
L'ottimizzazione rimane ancora un problema da risolvere. Ad esempio, un lavoro successivo sarà quello di cercare strutture dati più adatte delle liste a gestire le parole. Il problema, che ha impedito una veloce e facile soluzione, è che l'insieme di parole da gestire non è solo I, ma I e tutte le parole circolari via via generate, cioè un insieme che cresce dinamicamente con l'operazione di splicing.

ORGANIZZAZIONE DELLA TESI

Tale lavoro si sviluppa in sei capitoli, compreso il presente. Il secondo capitolo fornisce la definizione di base delle parole e parole circolari. Il terzo capitolo descrive brevemente il processo biologico(*cut and paste*), dello **splicing,** formalmente presentato nel quarto capitolo. Il quinto capitolo descrive l'implementazione delle parole circolari. Infine l'ultimo capitolo descrive l'implementazione dello Splicing circolare relativo al caso generale, e il tipo di ottimizzazione adottata