

Technical Communication for Computer Scientists  
Summer 2013

# Final Report: The Movie-Chain-Runner Project

June 26, 2013

Submitted to  
Thomas M. Keating  
Assistant Teaching Professor  
Computer Science Department  
Carnegie Mellon University

Prepared and Submitted by

Shashank Singh  
sss1@andrew.cmu.edu

Eugene Scanlon  
escanlon@cmu.edu

## Abstract

We attempted to solve the Movie-Chain-Runner Problem, a computational problem equivalent to the well-known Longest Path Problem. We designed and used Python to implement several algorithms for the problem, and include our best solution. In the end, we accomplished our original goal of constructing a movie chain of at least 300 titles. We also performed some analyses of both the problem and our algorithms, and use them here to discuss approaches to the problem. Finally, we overview and reflect on our group's approach and success.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	The Movie-Chain-Runner Problem . . . . .	1
1.2	Solution . . . . .	1
1.3	Did We Succeed? . . . . .	1
<b>2</b>	<b>Approach</b>	<b>2</b>
2.1	Programming Languages . . . . .	2
2.2	Algorithms . . . . .	2
2.3	Schedule . . . . .	3
<b>3</b>	<b>Results</b>	<b>3</b>
<b>4</b>	<b>Discussion</b>	<b>4</b>
4.1	Evaluation . . . . .	4
4.2	Lessons Learned . . . . .	4
4.3	Recomendations . . . . .	5
<b>5</b>	<b>Sources</b>	<b>5</b>
	<b>Appendices</b>	<b>6</b>
<b>A</b>	<b>Time Prediction</b>	<b>6</b>
<b>B</b>	<b>Our Longest Movie Chain</b>	<b>7</b>

# 1 Introduction

We first introduce the Movie-Chain-Runner Problem, our solution, and our success.

## 1.1 The Movie-Chain-Runner Problem

The Movie-Chain-Runner Problem is to find the longest chain of overlapping titles in a list of movie titles, where two titles are said to overlap if some suffix of the first movie is identical to some prefix of the second movie. For example, in the list

- Day of the Dead
- Live and Let Die
- Dead Poets' Society
- Die Another Day
- The Last Samurai

the longest chain consists of 4 titles:

Live and Let Die Another Day of the Dead Poets' Society

Our project is to approximately solve the Movie-Chain-Runner Problem for a given list of 6561 movie titles. By appropriately representing the movie list as a graph, we found that the Movie-Chain-Runner Problem is equivalent to the Longest Path Problem (finding the longest simple path in a directed graph). The Longest Path Problem is well known to be NP-Hard, meaning that no efficient algorithm exists to find longest paths in large graphs, including our movie graph. A review of the literature reveals that good approximate longest paths also cannot be found efficiently in large graphs.<sup>1</sup> Thus, our project is to study the movie title graph and innovate ways to find long paths in the graph.

## 1.2 Solution

We constructed our 301 title chain in four major steps. Our first step was to reconstruct the list of movie titles as a directed graph, thus reducing the problem to the well-known Longest Path Problem. The second step was to implement a simple brute-force algorithm. The final two steps involved augmenting this brute-force algorithm with two new procedures: “reversal” and “replacement.” The algorithms are described in detail under Section 2.3. on page 2.

## 1.3 Did We Succeed?

Our original goal was to surpass 300 titles. Despite some early setbacks leading us to lower our goal to 285 titles, we reached our original goal of 300 titles with 301 titles (see Appendix B for the full movie chain). We did not achieve the optional goal of 328 titles needed to set a new record.

---

<sup>1</sup>Björklund, Husfeldt, Khanna, “Approximating longest directed paths and cycles,” ICALP, 2004. (accessed June 26, 2013 a <http://citeseerx.ist.psu.edu/viewdoc/similar?doi=10.1.1.105.262>).

## 2 Approach

### 2.1 Programming Languages

Our team considered programming in Python, C, or MATLAB. We settled on Python because our entire team was familiar with Python, whereas only some team members had used C or MATLAB. Although C and MATLAB are faster languages, they are harder to use, and it would have taken too much time for team members to learn either. Jimmy also wrote Bash scripts to run our algorithms in parallel on cluster machines.

### 2.2 Algorithms

We present several algorithms (both successful and unsuccessful) that we considered.

#### Algorithms that worked:

Our first attempt at a solution was a brute-force algorithm which greedily followed every path in a depth-first search pattern, and retained the longest path found. We ran this algorithm until it ceased to make appreciable progress at a chain of 243 titles.

Our next step, “reversal,” was to reverse all edges in the movie graph and use the brute force algorithm to extend the beginning of the 243 title chain. We felt this approach would be useful because, upon running our original algorithm multiple times, we found that the beginning of the chain changed significantly, but end of the chain was always the same, suggesting that the end of the chain was already nearly optimal. Again, we ran the algorithm until it ceased to make appreciable progress, this time with a chain of 277 titles.

Our final step, “insertion,” checked if there was a longer subchain between two titles in our 277 title chain. Simply attempting to “squeeze” chains in between titles in our chain only increased our chain length to 278. However, when we allowed the insertion algorithm to replace entire subchains (“replacement”), we were able to construct our longest chain of 301 titles before the algorithm ceased to make much progress.

#### Algorithms that didn’t work:

We tried an algorithm that computes acyclic subgraphs of our graph and then runs a polynomial time Longest Path algorithm known for acyclic graphs. However, the number of cycles proved too large to generate all the subgraphs, and so we abandoned this effort.

We also considered some more elaborate algorithms we found in the literature: a color-coding algorithm proposed by Alon et al.<sup>2</sup> and the genetic algorithms studied by Portugal et al.<sup>3</sup> However, due to the complexity of implementing these algorithms and the success of our simpler algorithms, we opted not to try them.

---

<sup>2</sup>Noga Alon, Raphael Yuster, Uri Zwick, Color-coding, *Journal of the ACM (JACM)*, v.42 n.4, p.844-856, July 1995. (accessed June 16, 2013 at <http://dl.acm.org/citation.cfm?id=210337>).

<sup>3</sup>D. Portugal, C. H. Antunes, R. Rocha, “A Study of Genetic Algorithms for Approximating the Longest Path in Generic Graphs,” *Proc. of the IEEE SMC*, pp. 2539-2544, 2010. (accessed June 16, 2013 at <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5641920&navigation=1>).

## 2.3 Schedule

Figure 1 below shows the original Gantt chart presented in our project proposal. Figure 2 below shows the revised Gantt chart we presented in our progress report. The only changes were that the 300 title goal originally set for June 25 was eliminated and the 285 title goal originally set for June 17 was extended to June 25. In practice, the only deviation from our original schedule was achieving our 285 title goal one day late (June 18), but we caught up by reaching our 300 title goal the next day (June 19).

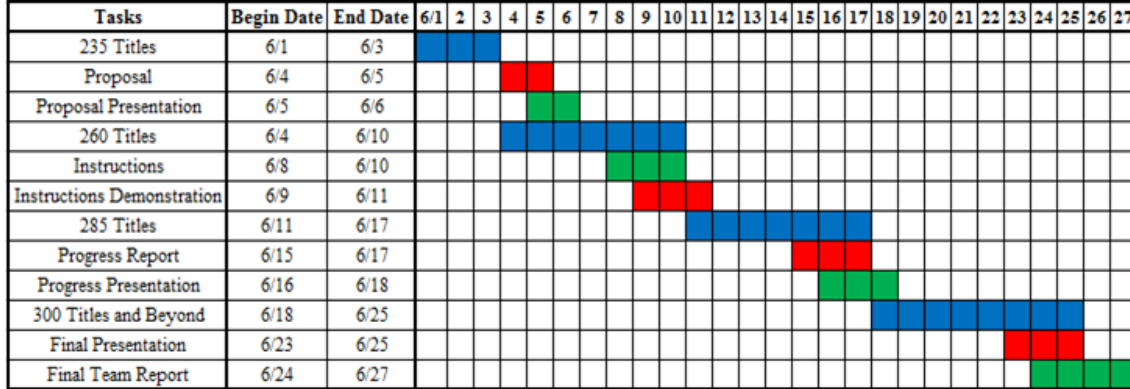


Figure 1: Our original Gantt chart.

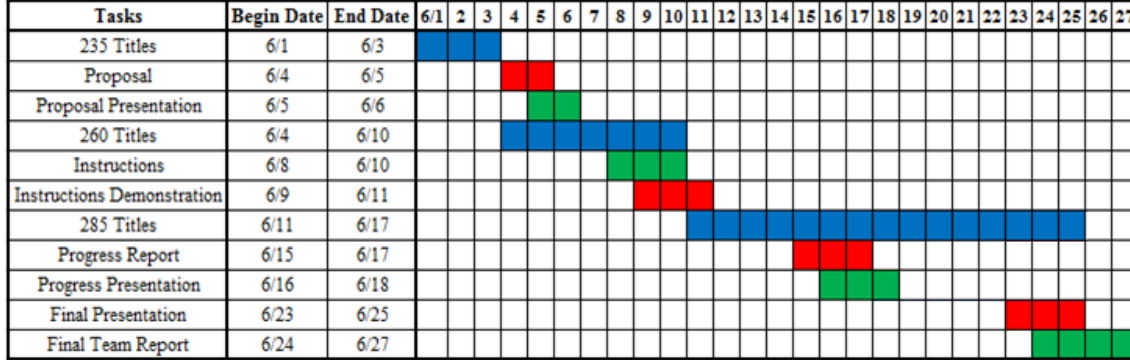


Figure 2: Our revised Gantt chart.

Key:	
Eugene/Sung	
Shashank/Yiming	
Everyone	

## 3 Results

We succeeded in finding a movie chain of 301 titles, fulfilling our scheduled goal of having a 300+ title chain by June 25.

Algorithm	Longest Chain
Brute-Force	243
Brute-Force w/ reversal	277
Brute-Force w/ reversal and insertion	278
Brute-Force w/ reversal and replacement	301

Table 1: Longest chain found by each algorithm

## 4 Discussion

Here we present the project evaluation promised in our proposal, followed by a discussion of lessons learned from the project and the resulting recommendations for future groups.

### 4.1 Evaluation

As suggested in our proposal, we evaluated the success of our project using three methods.

#### Method 1: Longest Chain Length

Our project was a success in that the number of titles in our longest chain exceeded 300. Although we failed to establish a new record, this secondary goal was optional and not expected.

#### Method 2: Algorithm Comparison

Our second method of evaluation was to compare different algorithms to decide on a “best” algorithm for the problem. Although we did not test a diverse selection of algorithms, we did try several variants on the basic brute force algorithm (as explained on page 2, under Section 2.3. Table 1 above presents the longest chain found by each variant. It is clear from the results that both reversal and replacement are important augmentations of the basic brute-force algorithm.

#### Method 3: Time Prediction

In order to gauge the difficulty of the problem at hand, we also attempted to predict the time needed to solve the problem by exploring all possible paths in the movie graph. To do this, we first measured the time taken to solve the problem on many random subgraphs of the movie graph and used this data to fit an exponential relation between the size of the graph and the time taken for the computation. We then extrapolated this relation to conclude that performing the computation on the entire graph would take about 1.03 trillion years, approximately 100 times the lifespan of the sun. Appendix A gives a detailed discussion of the experiment and its results and validity.

### 4.2 Lessons Learned

We learned several things about programming for large computational problems:

- Using primarily brute-force algorithms, we came quite close to (likely within 13% of) the optimal solution. Thus, although far from optimal in theory, brute-force algorithms can produce decent solutions for some NP-hard problems in practice.

- For algorithms with large asymptotic runtimes, constant factors are important - a 50% runtime reduction is very significant when the algorithm can take a day or more to run. This fact is really understated in CS classes.
- With a high level language (like Python) it is crucial to consider all available data structures and their implementations. We noticed too late that using a set rather than a list for part of our code might have sped up our program by a factor of 100.

### 4.3 Recomendations

Based on our experience, we strongly recommend attempting to implement simple algorithms first when tackling an NP-hard problem. While much of the literature is focused on optimizing asymptotic performance, in practice, a well implemented brute-force algorithm may perform nearly optimally with the time and hardware available, while being much simpler to implement correctly. Consequently, we also recommend trying to fully optimize code for such an implementation fairly early, since much time can be wasted in running the original program when one will later run a significantly optimized version.

Our team's work ethic and organization were crucial to our success. We recommend having frequent meetings (at least twice weekly) in order to coordinate work and ensure team members remain engaged.

## 5 Sources

- Git repository hosted on GitHub (accessed June 4, 2013)
  - <https://github.com/>
- Python Programming Language (accessed June 4, 2013)
  - <http://www.python.org/>

# Appendices

## A Time Prediction

To gauge the difficulty of the problem at hand, we predicted the time needed to solve the problem by exploring all possible paths in the movie graph. For each  $k \in \{100, 200, 300, \dots, 2500\}$ , we first generated 100 random subgraphs, each induced by  $k$  titles chosen uniformly at random. We then measured the runtime on each subgraph, and averaged over samples to estimate the runtime for  $k$  titles. We fitted an exponential relation between the size of the graph and the runtime (the parameters of the exponential are included in table 3). Evaluating this exponential function at  $x = 6561$  gives an estimate of **1.03 trillion years** needed to compute the solution for the entire 6561 title movie graph.

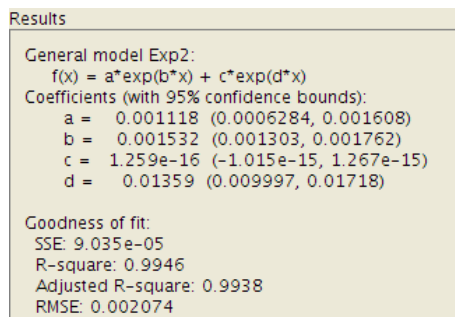


Figure 3: Exponential function from graph size to runtime

Figure 4 shows the fitted plot of how the runtime scales with the size of movie graph.

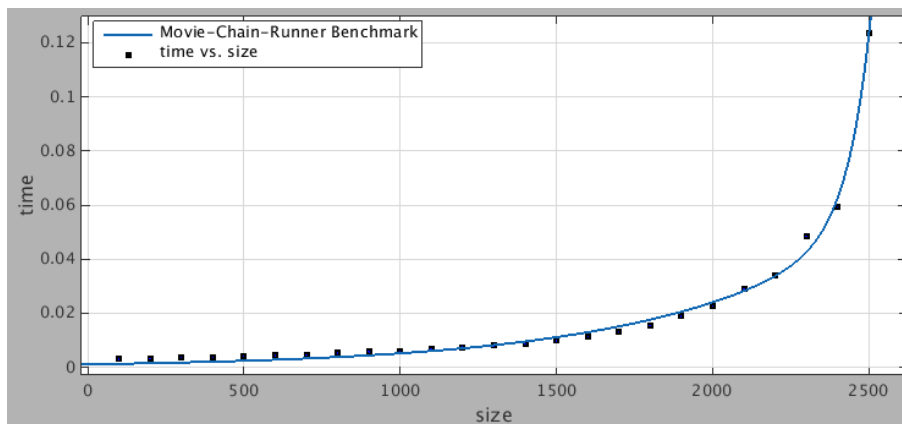


Figure 4: Plot of runtime (in seconds) over graph size (in titles).

The high  $R^2$  value suggests that our data fit the exponential relation quite well. Since we measured runtime rather than operations performed, the results depend on our programming language and computer. The tests were run sequentially in Python on a single GHC cluster machine. However, a powerful computer running C would reduce the runtime by at most a factor of 10,000, and so the computation would still take millions of years.



## B Our Longest Movie Chain

Here, we include our current longest movie chain, with 301 titles and 961 words:

THE GOSPEL	FEAR STRIKES OUT
THE GOSPEL OF JOHN	OUT OF THE PAST
JOHN Q	PAST MIDNIGHT
Q AND A	MIDNIGHT RUN
A PERFECT MURDER	RUN SILENT RUN DEEP
MURDER AND MURDER	DEEP BLUE
MURDER IN THE FIRST	DEEP BLUE SEA
FIRST BLOOD	SEA OF LOVE
BLOOD DIAMOND	LOVE HAPPY
DIAMOND MEN	HAPPY BIRTHDAY TO ME
MEN WITH GUNS	ME MYSELF I
GUNS OF THE MAGNIFICENT SEVEN	I SPY
SEVEN YEARS IN TIBET	SPY HARD
TIBET CRY OF THE SNOW LION	HARD TIMES
LION OF THE DESERT	TIMES SQUARE
DESERT BLUE	SQUARE DANCE
BLUE CAR	DANCE WITH A STRANGER
CAR 54 WHERE ARE YOU	STRANGER IN THE HOUSE
YOU CANT TAKE IT WITH YOU	THE HOUSE OF THE DEAD
YOU LIGHT UP MY LIFE	DEAD BANG
LIFE WITH FATHER	BANG BANG YOURE DEAD
FATHER OF THE BRIDE	DEAD END
BRIDE OF THE MONSTER	END OF DAYS
MONSTER HOUSE	DAYS OF HEAVEN
HOUSE OF DRACULA	HEAVEN CAN WAIT
DRACULA DEAD AND LOVING IT	WAIT UNTIL DARK
IT TAKES TWO	DARK CITY
TWO OR THREE THINGS I KNOW ABOUT HER	CITY OF JOY
HER MAJESTY MRS BROWN	JOY RIDE
BROWN SUGAR	RIDE THE HIGH COUNTRY
SUGAR AND SPICE	COUNTRY LIFE
SPICE WORLD	LIFE IS BEAUTIFUL
WORLD TRADE CENTER	BEAUTIFUL GIRLS
CENTER STAGE	GIRLS GIRLS GIRLS
STAGE FRIGHT	GIRLS JUST WANT TO HAVE FUN
FRIGHT NIGHT	FUN AND FANCY FREE
NIGHT FALLS ON MANHATTAN	FREE WILLY
MANHATTAN MURDER MYSTERY	FREE WILLY 2 THE ADVENTURE HOME
MYSTERY ALASKA	HOME ALONE
ALASKA SPIRIT OF THE WILD	ALONE IN THE DARK
THE WILD ONE	DARK STAR
ONE NIGHT STAND	STAR WARS EPISODE V THE EMPIRE STRIKES BACK
STAND IN	BACK TO THE BEACH
IN OLD CALIFORNIA	BEACH PARTY
CALIFORNIA SPLIT	PARTY GIRL
SPLIT SECOND	GIRL IN THE CADILLAC
SECOND BEST	CADILLAC MAN
BEST OF THE BEST	MAN OF THE HOUSE
THE BEST OF EVERYTHING	HOUSE OF FRANKENSTEIN
EVERYTHING RELATIVE	FRANKENSTEIN AND THE MONSTER FROM HELL
RELATIVE FEAR	HELL NIGHT

NIGHT AND DAY  
DAY FOR NIGHT  
NIGHT AND THE CITY  
CITY OF ANGELS  
ANGELS WITH DIRTY FACES  
FACES OF DEATH  
DEATH SHIP  
SHIP OF FOOLS  
FOOLS RUSH IN  
IN COLD BLOOD  
BLOOD SIMPLE  
SIMPLE MEN  
MEN IN BLACK  
BLACK AND WHITE  
WHITE LIGHTNING  
LIGHTNING IN A BOTTLE  
BOTTLE ROCKET  
ROCKET MAN  
MAN ON FIRE  
FIRE IN THE SKY  
SKY HIGH  
HIGH CRIMES  
CRIMES OF PASSION  
PASSION IN THE DESERT  
DESERT HEARTS  
HEARTS OF DARKNESS A FILMMAKERS APOCALYPSE  
APOCALYPSE NOW  
NOW YOU SEE HIM NOW YOU DONT  
DONT BOTHER TO KNOCK  
KNOCK OFF  
OFF THE BLACK  
THE BLACK ANGEL  
ANGEL HEART  
HEART CONDITION  
CONDITION RED  
RED DAWN  
DAWN OF THE DEAD  
DEAD HEAT  
HEAT AND DUST  
DUST TO GLORY  
GLORY ROAD  
ROAD HOUSE  
HOUSE PARTY  
PARTY MONSTER  
MONSTER IN A BOX  
BOX OF MOON LIGHT  
LIGHT IT UP  
UP CLOSE AND PERSONAL  
PERSONAL BEST  
BEST MEN  
MEN DONT LEAVE  
LEAVE HER TO HEAVEN  
HEAVEN AND EARTH  
EARTH GIRLS ARE EASY  
EASY MONEY

MONEY FOR NOTHING  
NOTHING BUT TROUBLE  
TROUBLE IN PARADISE  
PARADISE ROAD  
ROAD GAMES  
GAMES PEOPLE PLAY NEW YORK  
NEW YORK NEW YORK  
NEW YORK COP  
COP LAND  
LAND OF THE DEAD  
DEAD MAN  
DEAD MAN ON CAMPUS  
CAMPUS MAN  
MAN TROUBLE  
TROUBLE EVERY DAY  
DAY OF THE DEAD  
DEAD OF NIGHT  
NIGHT MOTHER  
MOTHER JUGS AND SPEED  
SPEED 2 CRUISE CONTROL  
CONTROL ROOM  
ROOM AT THE TOP  
TOP GUN  
GUN CRAZY  
CRAZY AS HELL  
HELL UP IN HARLEM  
HARLEM RIVER DRIVE  
DRIVE ME CRAZY  
CRAZY PEOPLE  
PEOPLE I KNOW  
I KNOW WHAT YOU DID LAST SUMMER  
SUMMER CATCH  
CATCH A FIRE  
FIRE ON THE MOUNTAIN  
THE MOUNTAIN MEN  
MEN CRY BULLETS  
BULLETS OVER BROADWAY  
BROADWAY DANNY ROSE  
ROSE RED  
RED EYE  
EYE FOR AN EYE  
AN EYE FOR AN EYE  
EYE OF GOD  
GOD TOLD ME TO  
TO DIE FOR  
FOR YOUR EYES ONLY  
ONLY THE STRONG  
ONLY THE STRONG SURVIVE A CELEBRATION OF SOUL  
SOUL FOOD  
FOOD OF LOVE  
LOVE IS THE DEVIL  
THE DEVIL RIDES OUT  
OUT COLD  
COLD FEVER  
FEVER PITCH

PITCH BLACK  
BLACK HAWK DOWN  
DOWN WITH LOVE  
LOVE LIFE  
LIFE OR SOMETHING LIKE IT  
IT HAPPENED AT THE WORLDS FAIR  
FAIR GAME  
GAME OF DEATH  
DEATH WISH V THE FACE OF DEATH  
DEATH WISH  
WISH UPON A STAR  
A STAR IS BORN  
BORN AMERICAN  
AMERICAN HISTORY X  
X THE MAN WITH THE X RAY EYES  
EYES OF AN ANGEL  
ANGEL BABY  
BABY SECRET OF THE LOST LEGEND  
LEGEND OF THE LOST  
THE LOST BOYS  
BOYS AND GIRLS  
GIRLS WILL BE GIRLS  
GIRLS OF SUMMER  
SUMMER LOVERS  
LOVERS AND OTHER STRANGERS  
STRANGERS WHEN WE MEET  
MEET JOE BLACK  
BLACK LIKE ME  
ME WITHOUT YOU  
YOU ONLY LIVE ONCE  
ONCE AROUND  
AROUND THE BEND  
BEND OF THE RIVER  
THE RIVER WILD  
WILD THINGS  
THINGS TO COME  
COME AND GET IT  
IT HAPPENED ONE NIGHT  
ONE NIGHT WITH THE KING  
THE KING AND I  
I WANT TO LIVE  
LIVE AND LET DIE  
DIE MOMMIE DIE  
DIE MONSTER DIE  
DIE HARD

HARD EIGHT  
EIGHT AND A HALF WOMEN  
WOMEN IN LOVE  
IN LOVE AND WAR  
WAR OF THE WORLDS  
THE WORLDS FASTEST INDIAN  
INDIAN SUMMER  
SUMMER SCHOOL  
SCHOOL OF ROCK  
ROCK STAR  
STAR TREK THE MOTION PICTURE  
PICTURE BRIDE  
BRIDE OF THE WIND  
THE WIND AND THE LION  
THE LION KING  
KING OF THE JUNGLE  
JUNGLE 2 JUNGLE  
JUNGLE BOOK  
BOOK OF LOVE  
LOVE WALKED IN  
IN GODS HANDS  
HANDS ON A HARD BODY  
BODY DOUBLE  
DOUBLE TEAM  
TEAM AMERICA WORLD POLICE  
POLICE ACADEMY  
POLICE ACADEMY 3 BACK IN TRAINING  
TRAINING DAY  
DAY OF THE WOMAN  
THE WOMAN IN RED  
RED RIVER  
RIVER OF NO RETURN  
RETURN TO HORROR HIGH  
HIGH SCHOOL HIGH  
HIGH SPIRITS  
SPIRITS OF THE DEAD  
DEAD MAN WALKING  
WALKING AND TALKING  
TALKING ABOUT SEX  
SEX AND THE OTHER MAN  
MAN OF THE YEAR  
YEAR OF THE DRAGON  
DRAGON SEED  
SEED OF CHUCKY